aws

Leitfaden

AWS Identitäts- und Zugriffsverwaltung

AWS Identitäts- und Zugriffsverwaltung: Leitfaden

Table of Contents

Was ist IAM?

Follow us on Twitter

AWS Identity and Access Management (IAM) ist ein Webservice, mit dem Sie den Zugriff auf Ressourcen sicher kontrollieren können. AWS Mit IAM können Sie Berechtigungen zentral verwalten, mit denen gesteuert wird, auf welche AWS Ressourcen Benutzer zugreifen können. Sie verwenden IAM, um zu steuern, wer authentifiziert (angemeldet) und autorisiert (Berechtigungen besitzt) ist, Ressourcen zu nutzen.

Wenn Sie eine 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 Stammbenutzer-Anmeldeinformationen erfordern.

Inhalt

- Verwenden von IAM mit einem SDK AWS

Videoeinführung in IAM

AWS Training and Certification bietet eine 10-minütige Videoeinführung in IAM:

Einführung in AWS Identity and Access Management

IAM-Features

IAM bietet Ihnen die folgenden Features:

Gemeinsamer Zugriff auf Ihre AWS-Konto

Sie können anderen Benutzern die Berechtigung zum Verwalten und Verwenden von Ressourcen in Ihrem AWS -Konto erteilen, ohne dass Sie Ihr Passwort oder Ihren Zugriffsschlüssel freigeben müssen.

Differenzierte Berechtigungen

Sie können verschiedenen Benutzern verschiedene Berechtigungen für verschiedene Ressourcen erteilen. Sie könnten beispielsweise einigen Benutzern vollständigen Zugriff auf Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift und andere Dienste gewähren. AWS Anderen Benutzern können Sie Lesezugriff auf nur einige S3-Buckets, die Berechtigung zum Verwalten von nur einigen EC2-Instances oder den Zugriff auf Ihre Abrechnungsdaten gewähren, aber nichts anderes.

Sicherer Zugriff auf AWS Ressourcen für Anwendungen, die auf Amazon EC2 ausgeführt werden

Sie können IAM-Features verwenden, um Anmeldeinformationen für Anwendungen, die auf EC2-Instances ausgeführt werden, sicher bereitzustellen. Diese Anmeldeinformationen berechtigen Ihre Anwendung zum Zugriff auf andere AWS Ressourcen. Beispiele hierfür sind S3-Buckets und DynamoDB-Tabellen.

Multi-Faktor-Authentifizierung (MFA)

Sie können eine Zwei-Faktor-Authentifizierung zu Ihrem Konto und einzelnen Benutzern hinzufügen, um mehr Sicherheit zu erhalten. Mit MFA müssen Sie oder Ihre Benutzer nicht nur ein Passwort oder einen Zugriffsschlüssel angeben, um mit Ihrem Konto zu arbeiten, sondern auch einen Code eines speziell konfigurierten Geräts. Wenn Sie bereits einen

FIDO-Sicherheitsschlüssel mit anderen Diensten verwenden und dieser über eine AWS unterstützte Konfiguration verfügt, können Sie ihn WebAuthn für die MFA-Sicherheit verwenden. Weitere Informationen finden Sie unter Unterstützte Konfigurationen für die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln.

Identitätsverbund

Sie können Benutzern, die bereits an anderer Stelle über ein Passwort verfügen – z. B. in Ihrem Unternehmensnetzwerk oder bei einem Internet-Identitätsanbieter – vorübergehend Zugang zu Ihrem AWS-Konto gewähren.

Identitätsinformationen für Zusicherung

Wenn Sie AWS CloudTrail verwenden, erhalten Sie Protokolldatensätze mit Informationen zu den Benutzern, die Anforderungen nach Ressourcen in Ihrem Konto gestellt haben. Diese Informationen basieren auf IAM-Identitäten.

Compliance mit PCI DSS

IAM unterstützt die Verarbeitung, Speicherung und Übertragung von Kreditkartendaten durch einen Händler oder Dienstleister und wurde als konform mit dem Payment Card Industry (PCI) Data Security Standard (DSS) validiert. Weitere Informationen zu PCI DSS, einschließlich der Möglichkeit, eine Kopie des AWS PCI Compliance Package anzufordern, finden Sie unter PCI DSS Level 1.

In viele Dienste integriert AWS

Eine Liste der AWS Dienste, die mit IAM funktionieren, finden Sie unterAWS Dienste, die mit IAM funktionieren.

Eventually Consistent-Konzept

IAM ist, wie viele andere AWS Dienste, letztlich konsistent. IAM erreicht eine hohe Verfügbarkeit, indem die Daten innerhalb der weltweit verteilten Amazon-Rechenzentren über mehrere Server repliziert werden. Wenn eine Anforderung zur Änderung von Daten erfolgreich ist, wird die Änderung übernommen und sicher gespeichert. Allerdings muss die Änderung in IAM repliziert werden, was einige Zeit dauern kann. Solche Änderungen umfassen das Erstellen oder Aktualisieren von Benutzern, Gruppen, Rollen und Richtlinien. Wir empfehlen, dass Sie in den kritischen, hochverfügbaren Code-Pfaden Ihrer Anwendung keine solchen IAM-Änderungen vornehmen. Nehmen Sie IAM-Änderungen stattdessen in einer separaten Initialisierungs- oder Einrichtungsroutine vor, die seltener ausgeführt wird. Vergewissern Sie sich auch, dass die Änderungen weitergegeben wurden, bevor die Produktionsarbeitsabläufe davon abhängen.

Weitere Informationen finden Sie unter Änderungen, die ich vornehme, sind nicht immer direkt sichtbar.

Kostenlos

AWS Identity and Access Management (IAM) und AWS Security Token Service (AWS STS) sind Funktionen Ihres AWS Kontos, die ohne zusätzliche Kosten angeboten werden. Es fallen nur Gebühren an, wenn Sie mit Ihren IAM-Benutzern oder AWS STS temporären Sicherheitsanmeldedaten auf andere AWS Dienste zugreifen. Informationen zu den Preisen anderer AWS Produkte finden Sie auf der Preisseite von Amazon Web Services.

Zugriff auf IAM

Sie können auf jede AWS Identity and Access Management der folgenden Arten damit arbeiten.

AWS Management Console

Die Konsole ist eine browserbasierte Oberfläche zur Verwaltung von IAM und AWS Ressourcen. Weitere Informationen zum Zugriff auf IAM über die Konsole finden Sie im AWS-Anmeldung - Benutzerhandbuch unter So melden Sie sich bei AWS an.

AWS Befehlszeilentools

Sie können die AWS Befehlszeilentools verwenden, um Befehle an der Befehlszeile Ihres Systems auszugeben, um IAM und AWS Aufgaben auszuführen. Die Verwendung der Kommandozeile kann schneller und bequemer sein als die Konsole. Die Befehlszeilentools sind auch nützlich, wenn Sie Skripts erstellen möchten, die AWS Aufgaben ausführen.

AWS stellt zwei Gruppen von Befehlszeilentools bereit: das AWS Command Line Interface(AWS CLI) und das AWS Tools for Windows PowerShell. Informationen zur Installation und Verwendung von finden Sie im AWS Command Line Interface Benutzerhandbuch. AWS CLI Informationen zur Installation und Verwendung der Tools für Windows PowerShell finden Sie im AWS Tools for Windows PowerShell Benutzerhandbuch.

Nachdem Sie sich bei der Konsole angemeldet haben, können Sie CLI- oder SDK-Befehle AWS CloudShell von Ihrem Browser aus ausführen. Die Berechtigungen für den Zugriff auf AWS Ressourcen basieren auf den Anmeldeinformationen, mit denen Sie sich bei der Konsole angemeldet haben. Abhängig von Ihrer Erfahrung ist die CLI möglicherweise eine effizientere Methode zur Verwaltung Ihres AWS-Konto. Weitere Informationen finden Sie unter Wird verwendet AWS CloudShell , um mit AWS Identity and Access Management zu arbeiten.

AWS SDKs

AWS bietet SDKs (Software Development Kits), die aus Bibliotheken und Beispielcode für verschiedene Programmiersprachen und Plattformen (Java, Python, Ruby, .NET, iOS, Android usw.) bestehen. Die SDKs bieten eine bequeme Möglichkeit, programmatischen Zugriff auf IAM und zu erstellen. AWS Mithilfe der SDKs lassen sich unter anderem Anforderungen kryptografisch signieren, Fehler verwalten und Anforderungen automatisch wiederholen. Informationen zu den AWS SDKs, einschließlich deren Download und Installation, finden Sie auf der Seite Tools für Amazon Web Services.

IAM-Query-API

Sie können mithilfe der IAM-Abfrage-API AWS programmgesteuert auf IAM zugreifen, sodass Sie HTTPS-Anfragen direkt an den Service senden können. Wenn Sie die Abfrage-API nutzen, müssen Sie Code zur digitalen Signierung von Anforderungen mittels Ihrer Anmeldeinformationen einschließen. Weitere Informationen finden Sie unter Aufrufen der IAM-API mithilfe von HTTP-Abfrageanforderungen und der IAM-API-Referenz.

Wann verwende ich IAM?

Wenn Sie verschiedene berufliche Funktionen ausüben

AWS Identity and Access Management ist ein zentraler Infrastrukturdienst, der die Grundlage für die Zugriffskontrolle auf der Grundlage der darin enthaltenen Identitäten bildet. AWS Sie verwenden IAM jedes Mal, wenn Sie auf Ihr AWS -Konto zugreifen.

Wie Sie IAM verwenden, unterscheidet sich je nach Ihrer Arbeit in AWS.

- Servicebenutzer – Wenn Sie den AWS -Service zur Ausführung von Aufgaben verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen bereit, die Sie benötigen. Wenn Sie für Ihre Arbeit erweiterte Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Fuktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen.
- Dienstadministrator — Wenn Sie in Ihrem Unternehmen für eine AWS Ressource verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf IAM. Es ist Ihre Aufgabe, zu bestimmen, auf welche IAM-Funktionen und Ressourcen Ihre Service-Benutzer 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.

- IAM-Administrator – Wenn Sie ein IAM-Administrator sind, verwalten Sie IAM-Identitäten und schreiben Richtlinien, um den Zugriff auf IAM zu verwalten.

Wenn Sie berechtigt sind, auf AWS -Ressourcen zuzugreifen

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.

Wenn Sie sich als IAM-Benutzer anmelden.

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.

Wenn Sie eine IAM-Rolle annehmen

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.

- Anwendungen, die auf Amazon EC2 ausgeführt werden — 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.

Wenn Sie Richtlinien und Berechtigungen erstellen

Sie können einem Benutzer Berechtigungen erteilen, indem Sie eine Richtlinie erstellen, d. h. ein Dokument, in dem die Aktionen aufgeführt sind, die ein Benutzer ausführen kann, sowie die Ressourcen, auf die sich diese Aktionen auswirken können. Alle Aktionen oder Ressourcen, die nicht explizit erlaubt sind, werden standardmäßig verweigert. Richtlinien können erstellt und an Prinzipale (Benutzer, Benutzergruppen, von Benutzern angenommene Rollen und Ressourcen) angehängt werden.

Diese Richtlinien werden mit einer IAM-Rolle verwendet:

- Vertrauensrichtlinie – Definiert, welche Prinzipale die Rolle annehmen können und unter welchen Bedingungen. Eine Vertrauensrichtlinie ist eine spezifische ressourcenbasierte Richtlinie für IAM-Rollen. Eine Rolle kann nur eine Vertrauensrichtlinie haben.
- Identitätsbasierte Richtlinien (intern und verwaltet) – Diese Richtlinien definieren die Berechtigungen, die der Benutzer der Rolle ausüben kann (oder denen die Ausführung verweigert wird) und für welche Ressourcen.

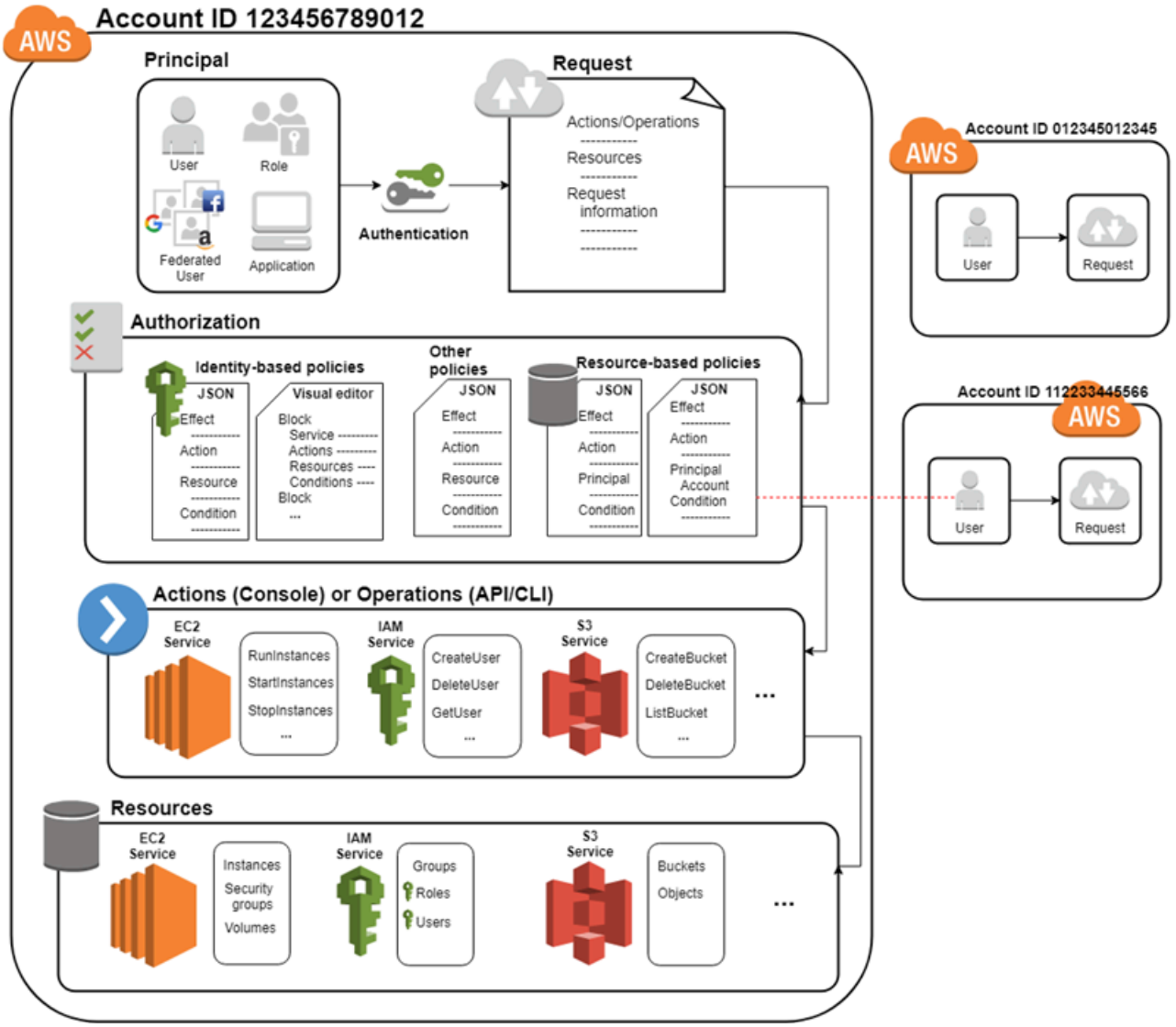
Verwenden Sie die Beispiele für identitätsbasierte Richtlinien in IAM, um Berechtigungen für Ihre IAM-Identitäten zu definieren. Nachdem Sie die benötigte Richtlinie gefunden haben, wählen Sie

„View the policy (Richtlinie anzeigen)“, um den JSON-Code der Richtlinie anzuzeigen. Sie können das JSON-Richtliniendokument als Vorlage für Ihre eigenen Richtlinien verwenden.

> **Note**
>
> Wenn Sie IAM Identity Center zur Verwaltung Ihrer Benutzer verwenden, weisen Sie in IAM Identity Center Berechtigungssätze zu, anstatt einem Prinzipal eine Berechtigungsrichtlinie zuzuweisen. Wenn Sie einer Gruppe oder einen Berechtigungssatz zuweisen Benutzer im AWS IAM Identity Center, erstellt IAM Identity Center entsprechende IAM-Rollen in jedem Konto und ordnet diesen Rollen die im Berechtigungssatz angegebenen Richtlinien zu. IAM Identity Center verwaltet die Rolle und ermöglicht es den autorisierten Benutzern, die Rolle anzunehmen. Wenn Sie den Berechtigungssatz ändern, stellt IAM Identity Center sicher, dass die entsprechenden IAM-Richtlinien und -Rollen entsprechend aktualisiert werden. Weitere Informationen zu IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

Funktionsweise von IAM

IAM stellt die notwendige Infrastruktur zur Verfügung, um die Authentifizierung und Autorisierung für Ihr AWS-Konto zu steuern. Das folgende Diagramm verdeutlicht diese IAM-Infrastruktur.



Zunächst verwendet ein menschlicher Benutzer oder eine Anwendung seine Anmeldeinformationen, um sich bei AWS zu authentifizieren. Die Authentifizierung erfolgt, indem die Anmeldeinformationen einem Prinzipal (einem IAM-Benutzer, Verbundbenutzer, IAM-Rolle oder Anwendung) zugeordnet werden, dem das AWS-Konto vertraut.

Als Nächstes wird eine Anfrage gestellt, um dem Prinzipal Zugriff auf Ressourcen zu gewähren. Der Zugriff wird als Antwort auf eine Autorisierungsanfrage gewährt. Wenn Sie sich beispielsweise zum ersten Mal bei der Konsole anmelden und sich auf der Startseite der Konsole befinden, greifen Sie nicht auf einen bestimmten Service zu. Wenn Sie einen Service auswählen, wird die Autorisierungsanfrage an diesen Service gesendet und es wird geprüft, ob Ihre Identität auf der Liste der autorisierten Benutzer steht, welche Richtlinien zur Kontrolle der gewährten Zugriffsebene durchgesetzt werden und welche anderen Richtlinien möglicherweise in Kraft sind.

Autorisierungsanfragen können von Schulleitern innerhalb Ihres Unternehmens AWS-Konto oder von einem anderen AWS-Konto , dem Sie vertrauen, gestellt werden.

Nach der Autorisierung kann der Prinzipal Maßnahmen ergreifen oder Operationen an Ressourcen in Ihrem AWS-Konto durchführen. Der Principal könnte beispielsweise eine neue Amazon Elastic Compute Cloud Instance starten, die IAM-Gruppenmitgliedschaft ändern oder Buckets löschen Amazon Simple Storage Service .

Grundkonzepte

- Bedingungen
- Auftraggeber
- Anforderung
- Authentifizierung
- Autorisierung
- Aktionen oder Operationen
- Ressourcen

Bedingungen

Diese IAM-Begriffe werden häufig verwendet, wenn Sie arbeiten mit: AWS

IAM-Ressource

IAM-Ressourcen werden in IAM gespeichert. Sie können sie in IAM hinzufügen, bearbeiten und entfernen.

- user
- Gruppe
- Rolle
- policy
- identity-provider object

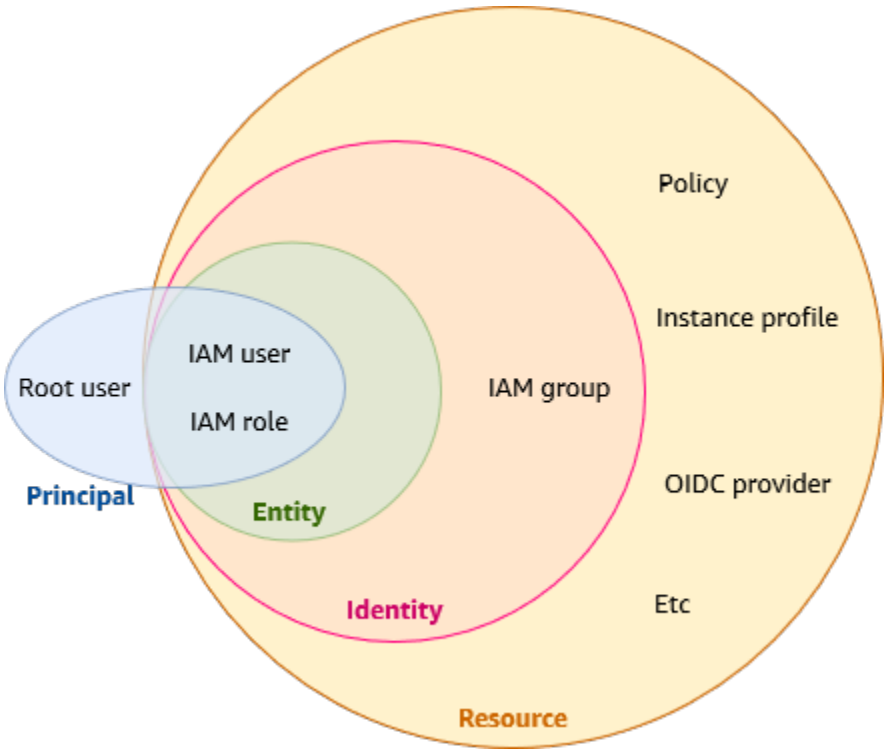
IAM-Entität

IAM-Ressourcen, die für die Authentifizierung AWS verwendet werden. Entitäten können in einer ressourcenbasierten Richtlinie als Prinzipal angegeben werden.

- user
- role

IAM-Identität

Eine IAM-Ressource, die in Richtlinien zur Durchführung von Aktionen und zum Zugriff auf Ressourcen autorisiert werden kann. Identitäten schließen Benutzer, Gruppen und Rollen ein.



Auftraggeber

Eine Person oder Anwendung, die die Root-Benutzer des AWS-Kontos, einen IAM-Benutzer oder eine IAM-Rolle verwendet, um sich anzumelden und Anfragen an diese zu richten. AWS Auftraggeber umfassen Verbundbenutzer und angenommene Rollen.

Menschliche Benutzer

Menschliche Benutzer, auch bekannt als menschliche Identitäten, sind die Personen, Administratoren, Entwickler, Betreiber und Verbraucher Ihrer Anwendungen.

Workload

Eine Sammlung von Ressourcen und Code, die einen geschäftlichen Nutzen erbringen, z. B. eine Anwendung oder ein Backend-Prozess. Kann Anwendungen, Betriebstools und Komponenten enthalten.

Auftraggeber

Ein Principal ist ein menschlicher Benutzer oder ein Workload, der eine Aktion oder einen Vorgang für eine Ressource anfordern kann. AWS Nach der Authentifizierung können dem Prinzipal je nach Prinzipaltyp entweder permanente oder temporäre Anmeldeinformationen für Anfragen erteilt werden. AWS IAM-Benutzern und Root-Benutzern werden permanente Anmeldeinformationen gewährt, während Rollen temporäre Anmeldeinformationen gewährt werden. Als bewährte Methode empfehlen wir, menschlichen Benutzern und Workloads den Zugriff auf AWS Ressourcen mithilfe temporärer Anmeldeinformationen vorzuschreiben.

Anforderung

Wenn ein Principal versucht AWS Management Console, die, die AWS API oder den zu verwenden AWS CLI, sendet dieser Principal eine Anfrage an AWS. Die Anforderung enthält die folgenden Informationen:

- Aktionen oder Operationen – Die Aktionen oder Operationen, die der Auftraggeber durchführen möchte. Dies kann eine Aktion in der AWS Management Console oder eine Operation in der AWS CLI oder AWS API sein.
- Ressourcen — Das AWS Ressourcenobjekt, für das die Aktionen oder Operationen ausgeführt werden.
- Auftraggeber – Die Person oder Anwendung, die eine Entität (Benutzer oder Rolle) verwendet hat, um die Anforderung zu senden. Informationen über den Auftraggeber beinhalten die Richtlinien, die der Entität zugeordnet sind, die der Auftraggeber zur Anmeldung verwendet hat.
- Umgebungsdaten – Informationen über die IP-Adresse, den Benutzeragent, den SSL-Status oder die Tageszeit.
- Ressourcendaten – Daten zu den angeforderten Ressourcen. Dies sind zum Beispiel Informationen wie ein DynamoDB-Tabellenname oder Tag auf einer Amazon EC2-Instance.

AWS fasst die Anforderungsinformationen in einem Anforderungskontext zusammen, der zur Auswertung und Autorisierung der Anfrage verwendet wird.

Authentifizierung

Ein Principal muss mit seinen Anmeldeinformationen authentifiziert (angemeldet AWS) werden, um eine Anfrage an ihn zu senden. AWS Einige Dienste, wie Amazon S3 und AWS STS, ermöglichen einige Anfragen von anonymen Benutzern. Sie stellen jedoch die Ausnahme von der Regel dar.

Um sich über die Konsole als Stammbenutzer zu authentifizieren, müssen Sie sich mit Ihrer E-Mail-Adresse und Ihrem Passwort anmelden. Als Verbundbenutzer werden Sie von Ihrem Identitätsanbieter authentifiziert und erhalten Zugriff auf AWS Ressourcen, indem Sie IAM-Rollen annehmen. Als IAM-Benutzer geben Sie Ihre Konto-ID oder den Alias und anschließend Ihren Benutzernamen und Ihr Passwort ein. Um Workloads über die API oder AWS CLI zu authentifizieren, können Sie temporäre Anmeldeinformationen verwenden, indem Sie eine Rolle zugewiesen bekommen, oder Sie können langfristige Anmeldeinformationen verwenden, indem Sie Ihren Zugriffsschlüssel und Ihren geheimen Schlüssel angeben. Möglicherweise müssen Sie auch zusätzliche Sicherheitsinformationen angeben. Als bewährte Methode AWS empfiehlt es sich, Multi-Faktor-Authentifizierung (MFA) und temporäre Anmeldeinformationen zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen zu den IAM-Entitäten, die sich authentifizieren AWS können, finden Sie unter und. IAM-Benutzer IAM-Rollen

Autorisierung

Sie müssen auch autorisiert sein (zulässig), um Ihre Anforderung abzuschließen. Während der Autorisierung verwendet AWS die Werte aus dem Anforderungskontext, um nach Richtlinien zu suchen, die auf die Anforderung anzuwenden sind. Anschließend wird anhand der Richtlinien festgelegt, ob die Anforderung zugelassen oder abgelehnt werden soll. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert und spezifizieren die Berechtigungen für Prinzipalentitäten. Es gibt mehrere Arten von Richtlinien, die sich darauf auswirken können, ob eine Anforderung zugelassen wird. Um Ihren Benutzern Berechtigungen für den Zugriff auf die AWS Ressourcen in ihrem eigenen Konto zu gewähren, benötigen Sie nur identitätsbasierte Richtlinien. Ressourcenbasierte Richtlinien werden häufig zum Erteilen von kontoübergreifenden Zugriff verwendet. Die anderen Richtlinientypen sind erweiterte Funktionen und sollten mit Bedacht eingesetzt werden.

AWS überprüft jede Richtlinie, die für den Kontext Ihrer Anfrage gilt. Wenn eine einzelne Berechtigungsrichtlinie eine verweigerte Aktion beinhaltet, AWS lehnt sie die gesamte Anfrage ab und beendet die Auswertung. Dieser Vorgang wird als explizite Zugriffsverweigerung bezeichnet. Da Anfragen standardmäßig abgelehnt werden, wird Ihre Anfrage nur AWS autorisiert, wenn jeder Teil Ihrer Anfrage gemäß den geltenden Berechtigungsrichtlinien zulässig ist. Die Auswertungslogik für eine Anforderung in einem einzelnen Konto folgt diesen allgemeinen Regeln:

- Standardmäßig werden alle Anforderungen verweigert. (Anforderungen, die mit Root-Benutzer des AWS-Kontos -Anmeldeinformationen für Ressourcen im Konto gesendet werden, werden im Allgemeinen erlaubt.)

- Eine explizite Zugriffserlaubnis in einer Berechtigungsrichtlinie (identitätsbasiert oder ressourcenbasiert) hat Vorrang vor diesem Standardwert.
- Das Vorhandensein einer Organisationen-SCP, einer IAM-Berechtigungsgrenze oder einer Sitzungsrichtlinie überschreibt die Zugangserlaubnis. Wenn eine oder mehrere dieser Richtlinienarten vorhanden sind, müssen sie alle die Anfrage zulassen. Andernfalls wird sie implizit verweigert.
- Eine explizite Zugriffsverweigerung überschreibt jede Zugriffserlaubnis in einer Richtlinie.

Weitere Informationen dazu, wie alle Arten von Richtlinien ausgewertet werden, finden Sie unter Auswertungslogik für Richtlinien. Wenn Sie eine Anforderung in einem anderen Konto initiieren müssen, muss eine Richtlinie in dem anderen Konto Ihnen den Zugriff auf die Ressource erlauben und die IAM-Entität, die Sie zum Erstellen der Anfrage verwenden, muss eine identitätsbasierte Richtlinie haben, die die Anforderung erlaubt.

Aktionen oder Operationen

AWS Genehmigt die Aktionen oder Vorgänge in Ihrer Anfrage, nachdem Ihre Anfrage authentifiziert und autorisiert wurde. Operationen werden von einem Service definiert und enthalten Dinge, die Sie mit einer Ressource machen können (z. B. Anzeigen, Anlegen, Bearbeiten und Löschen der Ressource). Zum Beispiel unterstützt IAM ca. 40 Aktionen für eine Benutzerressource, einschließlich der folgenden Aktionen:

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

Damit ein Auftraggeber eine Operation ausführen kann, müssen Sie die erforderlichen Aktionen in eine Richtlinie aufnehmen, die für den Auftraggeber oder die betroffene Ressource gilt. Eine Liste der Aktionen, Ressourcentypen und Bedingungsschlüssel, die von den einzelnen Diensten unterstützt werden, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Ressourcen

Nachdem die in Ihrer Anfrage enthaltenen Vorgänge AWS genehmigt wurden, können sie für die zugehörigen Ressourcen in Ihrem Konto ausgeführt werden. Eine Ressource ist ein Objekt, das

innerhalb eines Services existiert. Beispiele sind eine Amazon EC2-Instance, ein IAM-Benutzer und ein Amazon S3-Bucket. Der Service definiert eine Reihe von Aktionen, die für jede Ressource ausgeführt werden können. Wenn Sie eine Anforderung erstellen, um eine unabhängige Aktion für eine Ressource durchzuführen, wird diese Anforderung abgelehnt. Wenn Sie beispielsweise eine IAM-Rolle löschen möchten, aber eine IAM-Gruppenressource bereitstellen, schlägt die Anforderung fehl. AWS Servicetabellen, aus denen hervorgeht, welche Ressourcen von einer Aktion betroffen sind, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Überblick über das AWS Identitätsmanagement: Benutzer

Sie können bestimmten Benutzern Zugriff auf Ihre AWS-Konto Daten gewähren und ihnen spezifische Berechtigungen für den Zugriff auf Ihre AWS-Konto Ressourcen gewähren. Sie können sowohl IAM als auch verwenden AWS IAM Identity Center , um neue Benutzer zu erstellen oder bestehende Benutzer zu verbinden. AWS Der Hauptunterschied zwischen beiden besteht darin, dass IAM-Benutzern langfristige Anmeldeinformationen für Ihre AWS Ressourcen gewährt werden, während Benutzer in IAM Identity Center über temporäre Anmeldeinformationen verfügen, die bei jeder Anmeldung des Benutzers eingerichtet werden. AWS Es hat sich bewährt, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um mit temporären Anmeldeinformationen zuzugreifen, anstatt sie als IAM-Benutzer zu AWS verwenden. Ein primärer Verwendungszweck für IAM-Benutzer besteht darin, Workloads, die keine IAM-Rollen verwenden können, die Möglichkeit zu geben, mithilfe der API oder CLI programmatische Anfragen an AWS Dienste zu stellen.

Themen

- Nur erstmaliger Zugriff: Ihre Stammbenutzer-Anmeldeinformationen
- IAM-Benutzer und Benutzer in IAM Identity Center
- Erstellen eines Verbunds von vorhandenen Benutzern
- Zugriffskontrollmethoden

Nur erstmaliger Zugriff: Ihre Stammbenutzer-Anmeldeinformationen

Wenn Sie eine 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. Nur Service Control Policies (SCPs) in Organisationen können die Berechtigungen einschränken, die dem Stammbenutzer gewährt werden.

IAM-Benutzer und Benutzer in IAM Identity Center

IAM-Benutzer sind keine separaten Konten. Sie sind in Ihrem Konto inbegriffene Benutzer. Jeder Benutzer kann über sein eigenes Passwort für den Zugriff auf die AWS Management Console verfügen. Darüber hinaus können Sie individuelle Zugriffsschlüssel für jeden Benutzer erstellen, sodass der Benutzer programmgesteuerte Anforderungen an die Ressourcen in Ihrem Konto stellen kann.

IAM-Benutzern werden langfristige Anmeldeinformationen für Ihre AWS Ressourcen gewährt. Als bewährte Methode sollten Sie keine IAM-Benutzer mit langfristigen Anmeldeinformationen für Ihre menschlichen Benutzer erstellen. Erfordern Sie stattdessen, dass Ihre menschlichen Benutzer beim Zugriff AWS temporäre Anmeldeinformationen verwenden.

> **Note**
>
> Für Szenarien, in denen Sie IAM-Benutzer mit programmgesteuertem Zugriff und langfristigen Anmeldeinformationen benötigen, empfehlen wir Ihnen, die Zugriffsschlüssel bei Bedarf zu aktualisieren. Weitere Informationen finden Sie unter Aktualisierung der Zugriffsschlüssel.

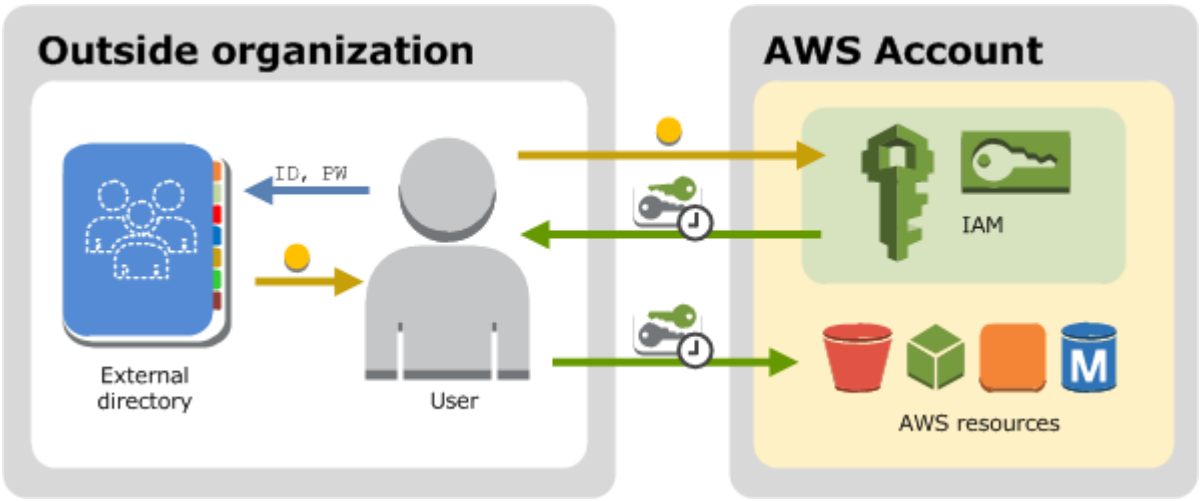
Im Gegensatz dazu Benutzer im AWS IAM Identity Center werden Ihnen kurzfristige Zugangsdaten für Ihre AWS Ressourcen gewährt. Für eine zentrale Zugriffsverwaltung empfehlen wir Ihnen die Verwendung von AWS IAM Identity Center (IAM Identity Center), um den Zugriff auf Ihre Konten und die Berechtigungen innerhalb dieser Konten zu verwalten. IAM Identity Center wird automatisch mit einem Identity Center-Verzeichnis als Standard-Identitätsquelle konfiguriert, in dem Sie Benutzer und Gruppen erstellen und deren Zugriffsebene Ihren AWS Ressourcen zuweisen können. Weitere Informationen finden Sie unter Was ist AWS IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

Erstellen eines Verbunds von vorhandenen Benutzern

Wenn die Benutzer in Ihrer Organisation bereits eine Möglichkeit haben, authentifiziert werden, z. B. indem sie sich bei Ihrem Unternehmensnetzwerk anmelden, müssen Sie keinen separaten

IAM-Benutzer oder Benutzer in IAM Identity Center für sie erstellen. Stattdessen können Sie diese Benutzeridentitäten so zusammenführen, dass sie entweder IAM oder AWS verwenden. AWS IAM Identity Center

Das folgende Diagramm zeigt, wie ein Benutzer temporäre AWS Sicherheitsanmeldeinformationen für den Zugriff auf Ressourcen in Ihrem abrufen kann. AWS-Konto



Der Verbund ist insbesondere in folgenden Fällen nützlich:

- Ihre Benutzer sind bereits in einem Unternehmensverzeichnis vorhanden.

  Wenn Ihr Unternehmensverzeichnis mit Security Assertion Markup Language 2.0 (SAML 2.0) kompatibel ist, können Sie Ihr Unternehmensverzeichnis so konfigurieren, dass es Ihren Benutzern Single-Sign-On-Zugriff (SSO) gewährt. AWS Management Console Weitere Informationen finden Sie unter Gängige Szenarien für temporäre Anmeldeinformationen.

  Wenn Ihr Unternehmensverzeichnis nicht mit SAML 2.0 kompatibel ist, können Sie eine Identity Broker-Anwendung erstellen, um Ihren Benutzern Single-Sign-On-Zugriff (SSO) auf die zu ermöglichen. AWS Management Console Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

  Wenn es sich bei Ihrem Unternehmensverzeichnis AWS IAM Identity Center um Microsoft Active Directory handelt, können Sie damit ein selbstverwaltetes Verzeichnis in Active Directory oder ein Verzeichnis in verbinden, AWS Directory Serviceum eine Vertrauensstellung zwischen Ihrem Unternehmensverzeichnis und Ihrem AWS-Konto herzustellen.

  Wenn Sie einen externen Identitätsanbieter (IdP) wie Okta oder Microsoft Entra verwenden AWS IAM Identity Center , um Benutzer zu verwalten, können Sie damit Vertrauen zwischen Ihrem IdP

und Ihrem herstellen. AWS-Konto Weitere Informationen finden Sie unter Verbinden mit einem externen Identitätsanbieter im AWS IAM Identity Center -Benutzerhandbuch.

- Ihre Benutzer verfügen bereits über Internet-Identitäten.

  Wenn Sie eine mobile oder webbasierte Anwendung erstellen, mit der sich Benutzer über einen Internet-Identitätsanbieter wie Login with Amazon, Facebook, Google oder einen beliebigen OpenID Connect (OIDC) kompatiblen Identitätsanbieter identifizieren können, kann die Anwendung Federation für den Zugriff auf AWS verwenden. Weitere Informationen finden Sie unter OIDC-Föderation.

  > **Tip**
  >
  > Um den Identitätsverbund mit Internet-Identitätsanbietern zu verwenden, empfehlen wir die Nutzung von Amazon Cognito.

Zugriffskontrollmethoden

So können Sie den Zugriff auf Ihre Ressourcen kontrollieren. AWS

Typ des Benutzerzugriffs	Verwendungsgrund	Wo erhalte ich weitere Informationen?
Single Sign-On (SSO)-Zugriff für menschliche Benutzer, z. B. Ihre Mitarbeiter, auf AWS -Ressourcen mithilfe von IAM Identity Center	Das IAM Identity Center bietet einen zentralen Ort, an dem die Verwaltung von Benutzern und deren Zugriff auf Cloud-Anwendungen sowie deren Zugriff AWS-Konten auf Cloud-Anwendungen zusammengeführt werden. Sie können einen Identitätsspeicher im IAM Identity Center einrichten oder den Verbund mit einem vorhanden	Weitere Informationen zum Einrichten von IAM Identity Center finden Sie unter Erste Schritte im AWS IAM Identity Center -Benutzerhandbuch. Weitere Informationen über MFA in IAM Identity Center finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center -Benutzerhandbuch.

Typ des Benutzerzugriffs	Verwendungsgrund	Wo erhalte ich weitere Informationen?
	en Identitätsanbieter (IdP) konfigurieren. Als bewährte Sicherheitsmethode wird empfohlen, Ihren menschlichen Benutzern begrenzte Zugangsdaten für AWS Ressourcen nach Bedarf zuzuweisen. Benutzer können sich einfacher anmelden, und Sie behalten die Kontrolle über ihren Zugriff auf Ressourcen von einem einzigen System aus. IAM Identity Center unterstützt die Multi-Faktor-Authentifizierung (MFA) für zusätzliche Kontosicherheit.	
Verbundzugriff für menschliche Benutzer, z. B. Benutzer Ihrer Belegschaft, auf AWS-Services, die IAM-Identitätsanbieter verwenden	IAM-Unterstützungen IdPs , die mit OpenID Connect (OIDC) oder SAML 2.0 (Security Assertion Markup Language 2.0) kompatibel sind. Nachdem Sie einen IAM-Identitätsanbieter erstellt haben, müssen Sie eine oder mehrere IAM-Rollen erstellen , die einem Verbundbenutzer dynamisch zugewiesen werden können.	Weitere Informationen zu IAM-Identitätsanbietern und Verbund finden Sie unter Identitätsanbieter und Verbund.

Typ des Benutzerzugriffs	Verwendungsgrund	Wo erhalte ich weitere Informationen?
Kontoüber greifender Zugriff zwischen AWS-Konten	Sie möchten den Zugriff auf bestimmte AWS Ressourcen mit Benutzern in anderen AWS-Konten teilen. Rollen stellen die primäre Möglichkeit dar, um kontoüber greifendem Zugriff zu gewähren. Einige AWS -Services ermöglichen Ihnen, eine Richtlinie direkt an eine Ressource anzufügen (anstatt eine Rolle als Proxy zu verwenden). Diese werden als ressourcenbasierte Richtlinien bezeichnet.	Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen. Weitere Informationen zu serviceverknüpften Rollen finden Sie unter Verwenden von serviceverknüpften Rollen. Informationen dazu, welche Services die Verwendung serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes (Ja) in der Spalte Serviceverknüpfte Rolle angegeben ist. Um die serviceverknüpfte Rollendokumentation für diesen Service anzuzeigen, wählen Sie den mit Ja verknüpften Link in der Spalte.

Typ des Benutzerzugriffs	Verwendungsgrund	Wo erhalte ich weitere Informationen?
Langfristige Anmeldeinformationen für bestimmte IAM-Benutzer in Ihrem AWS-Konto	Möglicherweise haben Sie spezielle Anwendungsfälle, für die langfristige Anmeldeinformationen mit registrierten IAM-Benutzern erforderlich sind. AWS Sie können IAM verwenden, um diese IAM-Benutzer in Ihrem zu erstellen AWS-Konto, und IAM verwenden, um ihre Berechtigungen zu verwalten. Einige der Anwendungsfälle umfassen Folgendes: • Workloads, die keine IAM-Rollen verwenden können. • AWS Clients von Drittanbietern, die programmatischen Zugriff über Zugriffsschlüssel benötigen • Servicespezifische Anmeldeinformationen für Amazon AWS CodeCommit Keyspaces • AWS IAM Identity Center ist für Ihr Konto nicht verfügbar und Sie haben keinen anderen Identitätsanbieter Als bewährte Methode in Szenarien, in denen Sie IAM-Benutzer mit programmg	Weitere Informationen zum Einrichten eines IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto. Weitere Informationen zu IAM-Benutzerzugriffsschlüsseln finden Sie unterVerwalten der Zugriffsschlüssel für IAM-Benutzer. Weitere Informationen zu dienstspezifischen Anmeldeinformationen für AWS CodeCommit Amazon Keyspaces finden Sie unter Verwenden von IAM mit CodeCommit: Git-Anmeldeinformationen, SSH-Schlüsseln und AWS Zugriffsschlüsseln und. Verwendung von IAM mit Amazon Keyspaces (für Apache Cassandra)

Typ des Benutzerzugriffs	Verwendungsgrund	Wo erhalte ich weitere Informationen?
	esteuertem Zugriff und langfristigen Anmeldeinformationen benötigen, empfehlen wir Ihnen, die Zugriffsschlüssel bei Bedarf zu aktualisieren. Weitere Informationen finden Sie unter Aktualisierung der Zugriffsschlüssel.	

Übersicht über die Zugriffsverwaltung: Berechtigungen und Richtlinien

Im Bereich Zugriffsverwaltung AWS Identity and Access Management (IAM) können Sie definieren, was eine Haupteinheit in einem Konto tun darf. Eine Auftraggeber-Entität ist eine Person oder eine Anwendung, die mithilfe einer IAM-Entität (Benutzer oder Rolle) authentifiziert wird. Zugriffsverwaltung wird oft als Autorisierung bezeichnet. Sie verwalten den Zugriff, AWS indem Sie Richtlinien erstellen und diese an IAM-Identitäten (Benutzer, Benutzergruppen oder Rollen) oder Ressourcen anhängen. AWS 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 Principal eine IAM-Entität (Benutzer oder Rolle) verwendet, um eine Anfrage zu stellen. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu diesen Richtlinienarten und ihrer Verwendung finden Sie unter Berechtigungen und Richtlinien in IAM.

Richtlinien und Konten

Wenn Sie ein einzelnes Konto in verwalten AWS, definieren Sie die Berechtigungen innerhalb dieses Kontos mithilfe von Richtlinien. Wenn Sie Berechtigungen über mehrere Konten hinweg verwalten, ist es schwieriger, Berechtigungen für Ihre Benutzer zu verwalten. Sie können IAM-Rollen, ressourcenbasierte Richtlinien oder Zugriffssteuerungslisten (Access Control Lists, ACLs) für kontoübergreifende Berechtigungen verwenden. Wenn Sie jedoch mehrere Konten besitzen,

empfehlen wir Ihnen stattdessen, den AWS Organizations Dienst zu nutzen, der Ihnen bei der Verwaltung dieser Berechtigungen hilft. Weitere Informationen finden Sie unter Was ist AWS Organizations? im Organizations User Guide.

Richtlinien und Benutzer

IAM-Benutzer sind Identitäten im Service. Wenn Sie einen IAM-Benutzer anlegen, kann er auf nichts in Ihrem Konto zugreifen, bis Sie ihm die Erlaubnis geben. Sie erteilen einem Benutzer Berechtigungen, indem Sie eine identitätsbasierte Richtlinie erstellen, bei der es sich um eine Richtlinie handelt, die mit dem Benutzer oder einer Gruppe, zu der der Benutzer gehört, verknüpft ist. Das folgende Beispiel zeigt eine JSON-Richtlinie, die dem Benutzer gestattet, alle Amazon DynamoDB-Aktionen (`dynamodb:*`) in der `Books`-Tabelle im `123456789012`-Konto in der Region `us-east-2` auszuführen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

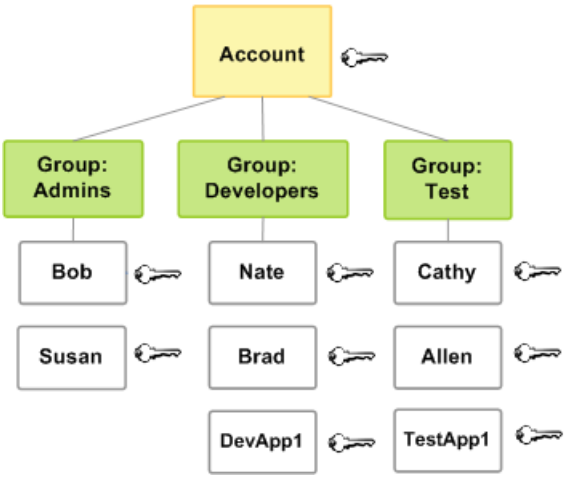
Nachdem Sie diese Richtlinie Ihrem IAM-Benutzer hinzugefügt haben, hat der Benutzer nur diese DynamoDB-Berechtigungen. Die meisten Benutzer haben mehrere Richtlinien, die gemeinsam die Berechtigungen für diesen Benutzer darstellen.

Aktionen oder Ressourcen, die nicht explizit erlaubt sind, werden standardmäßig verweigert. Wenn beispielsweise die oben genannte Richtlinie die einzige Richtlinie ist, die an einen Benutzer angefügt ist, darf dieser Benutzer nur DynamoDB-Aktionen in der Tabelle `Books` ausführen. Aktionen in allen anderen Tabellen sind verboten. Ebenso darf der Benutzer keine Aktionen in Amazon EC2, Amazon S3 oder einem anderen AWS Service ausführen. Der Grund dafür ist, dass Berechtigungen zum Arbeiten mit diesen Services nicht in der Richtlinie enthalten sind.

Richtlinien und Gruppen

Sie können IAM-Benutzer in IAM-Gruppen einteilen und einer Gruppe eine Richtlinie anfügen. In diesem Fall haben einzelne Benutzer weiterhin ihre eigenen Anmeldeinformationen, aber alle Benutzer in einer Gruppe verfügen über die Berechtigungen, die an die Gruppe angefügt sind.

Verwenden Sie Gruppen für eine einfachere Verwaltung von Berechtigungen und befolgen Sie unsere Bewährte Methoden für die Sicherheit in IAM.



Benutzern oder Gruppen können mehrere Richtlinien angefügt sein, die unterschiedliche Berechtigungen gewähren. In diesem Fall werden die Berechtigungen der Benutzer basierend auf der Kombination von Richtlinien berechnet. Aber das Grundprinzip gilt weiterhin: Wenn dem Benutzer keine explizite Berechtigung für eine Aktion und eine Ressource gewährt wurde, verfügt der Benutzer nicht über diese Berechtigungen.

Verbundbenutzer und -rollen

Verbundene Benutzer haben bei Ihnen keine dauerhaften Identitäten, wie dies bei IAM-Benutzern AWS-Konto der Fall ist. Um verbundenen Benutzern Berechtigungen zuzuweisen, können Sie eine Entität erstellen, die als Rolle bezeichnet wird, und Berechtigungen für die Rolle definieren. Wenn sich ein Verbundbenutzer anmeldet AWS, wird der Benutzer der Rolle zugeordnet und erhält die in der Rolle definierten Berechtigungen. Weitere Informationen finden Sie unter Erstellen von Rollen für externe Identitätsanbieter (Verbund).

Identitätsbasierte und ressourcenbasierte Richtlinien

Identitätsbasierte Richtlinien sind Berechtigungsrichtlinien, die Sie an eine IAM-Identität anfügen können, wie z. B. IAM-Benutzer, -Rollen oder -Gruppen. Ressourcenbasierte Richtlinien sind Berechtigungsrichtlinien, die Sie an eine Ressource wie z. B. einen Amazon S3-Bucket oder eine IAM-Rollenvertrauensrichtlinie anfügen.

Identitätsbasierte Richtlinien steuern, welche Aktionen die Identität für welche Ressourcen und unter welchen Bedingungen ausführen kann. Identitätsbasierte Richtlinien können weiter kategorisiert werden:

- Verwaltete Richtlinien — Eigenständige identitätsbasierte Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können. AWS-Konto Sie können zwei Arten von verwalteten Richtlinien verwalten:
  - AWS verwaltete Richtlinien — Verwaltete Richtlinien, die von erstellt und verwaltet werden. AWS Wenn Sie mit der Verwendung von Richtlinien noch nicht vertraut sind, empfehlen wir Ihnen, zunächst AWS verwaltete Richtlinien zu verwenden.
  - Vom Kunden verwaltete Richtlinien – Dies sind verwaltete Richtlinien, die Sie in Ihrem AWS-Konto erstellen und verwalten. Von Kunden verwaltete Richtlinien bieten eine genauere Kontrolle über Ihre Richtlinien als AWS verwaltete Richtlinien. Sie können eine IAM-Richtlinie im visuellen Editor oder durch direkte Erstellung des JSON-Richtliniendokuments erstellen, bearbeiten und validieren. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien und Bearbeiten von IAM-Richtlinien.
- Inline-Richtlinien – Dies sind Richtlinien, die Sie erstellen und verwalten und die Sie direkt in einen einzelnen Benutzer, einer Gruppe oder einer Rolle integrieren. In den meisten Fällen empfehlen wir nicht die Verwendung von Inline-Richtlinien.

Ressourcenbasierte Richtlinien steuern, welche Aktionen ein bestimmter Auftraggeber mit dieser Ressource durchführen kann und unter welchen Bedingungen dies möglich ist. Ressourcenbasierte Richtlinien sind Inline-Richtlinien. Es gibt keine verwalteten ressourcenbasierten Richtlinien. Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Auftraggeber in einer ressourcenbasierten Richtlinie angeben.

Der IAM-Service unterstützt nur eine Art von ressourcenbasierter Richtlinie, die als Rollen-Vertrauensrichtlinie bezeichnet wird, die einer IAM-Rolle zugewiesen ist. Da eine IAM-Rolle sowohl eine Identität als auch eine Ressource ist, die ressourcenbasierte Richtlinien unterstützt, müssen Sie sowohl eine Vertrauensrichtlinie als auch eine identitätsbasierte Richtlinie einer IAM-Rolle anfügen. Vertrauensrichtlinien definieren, welche Auftraggeber-Entitäten (Konten, Benutzer, Rollen und verbundene Benutzer) die Rolle übernehmen können. Informationen darüber, inwieweit sich IAM-Rollen von anderen ressourcenbasierten Richtlinien unterscheiden, finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.

Informationen darüber, welche Services ressourcenbasierte Richtlinien unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Weitere Informationen zu ressourcenbasierten Richtlinien finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien.

Wofür ist ABAC? AWS

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-Ressourcen, einschließlich IAM-Entitäten (Benutzer oder Rollen), und an AWS Ressourcen anhängen. Sie können eine einzelne ABAC-Richtlinie oder einen kleinen Richtliniensatz für Ihre IAM-Prinzipale erstellen. Diese ABAC-Richtlinien können so konzipiert werden, dass Operationen zugelassen werden, wenn das Tag des Prinzipals mit dem Ressourcen-Tag übereinstimmt. ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Sie können beispielsweise drei Rollen mit dem Tag-Schlüssel `access-project` erstellen. Legen Sie den Tag-Wert der ersten Rolle auf `Heart`, den zweiten auf `Star` und den dritten auf `Lightning` fest. Sie können dann eine einzelne Richtlinie verwenden, die den Zugriff erlaubt, wenn die Rolle und die Ressource mit demselben Wert für `access-project` markiert sind. Ein ausführliches Tutorial, das die Verwendung von ABAC in AWS demonstriert, finden Sie unter. IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren Weitere Informationen zu Services, die ABAC unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

Vergleich des ABAC-Modells mit dem herkömmlichen RBAC-Modell

Das herkömmliche in IAM verwendete Autorisierungsmodell wird als rollenbasierte Zugriffskontrolle (RBAC, Role-Based Access Control) bezeichnet. RBAC definiert Berechtigungen basierend auf der Job-Funktion einer Person, die außerhalb von AWS als Rollebezeichnet wird. Der Begriff „Innerhalb AWS einer Rolle“ bezieht sich normalerweise auf eine IAM-Rolle, bei der es sich um eine Identität in IAM handelt, von der Sie ausgehen können. IAM umfasst verwaltete Richtlinien für Job-Funktionen, die die Berechtigungen an eine Job-Funktion in einem RBAC-Modell anpassen.

In IAM implementieren Sie RBAC, indem Sie verschiedene Richtlinien für verschiedene Job-Funktionen erstellen. Anschließend fügen Sie die Richtlinien an Identitäten (IAM-Benutzer, -Benutzergruppen oder -IAM-Rollen) an. Als bewährte Methode erteilen Sie nur die für eine Auftragsfunktion minimal erforderlichen Berechtigungen. Dies wird als Gewähren von geringsten Rechten bezeichnet. Führen Sie dazu die spezifischen Ressourcen auf, auf die die Job-Funktion zugreifen kann. Der Nachteil des herkömmlichen RBAC-Modells besteht darin, dass Sie Richtlinien aktualisieren müssen, um Zugriff auf die Ressourcen zu gewähren, die von Mitarbeitern neu hinzugefügt werden.

Angenommen, Sie haben drei Projekte namens `Heart`, `Star` und `Lightning`, an denen Ihre Mitarbeiter arbeiten. Sie erstellen für jedes Projekt eine IAM-Rolle. Anschließend fügen Sie Richtlinien an die einzelnen IAM-Rollen an, um die Ressourcen festzulegen, auf die jeder, der die Rolle annehmen darf, zugreifen kann. Wenn ein Mitarbeiter Jobs innerhalb des Unternehmens wechselt, weisen Sie ihm eine andere IAM-Rolle zu. Personen oder Programme können mehreren Rollen zugewiesen werden. Für das `Star`-Projekt sind jedoch möglicherweise zusätzliche Ressourcen erforderlich, beispielsweise ein neuer Amazon-EC2-Container. In diesem Fall müssen Sie die der `Star`-Rolle zugeordnete Richtlinie aktualisieren, um die neue Containerressource anzugeben. Andernfalls dürfen `Star`-Projektmitglieder nicht auf den neuen Container zugreifen.

ABAC hat gegenüber dem herkömmlichen RBAC-Modell folgende Vorteile:

- ABAC-Berechtigungen lassen sich einfach an Innovationen anpassen. Es ist nicht mehr notwendig, dass ein Administrator vorhandene Richtlinien aktualisiert, um den Zugriff auf neue Ressourcen zu erlauben. Angenommen, Sie haben Ihre ABAC-Strategie mit dem `access-project`-Tag entwickelt. Ein Entwickler verwendet die Rolle mit dem `access-project` = `Heart`-Tag. Wenn Personen des `Heart`-Projekts zusätzliche Amazon EC2-Ressourcen benötigen, kann der Entwickler neue Amazon EC2-Instances mit dem `access-project` = `Heart`-Tag erstellen. Anschließend kann jeder Benutzer des `Heart`-Projekts diese Instances starten und stoppen, da die Tag-Werte übereinstimmen.
- ABAC erfordert weniger Richtlinien. Da Sie keine verschiedenen Richtlinien für verschiedene Job-Funktionen erstellen müssen, erstellen Sie insgesamt weniger Richtlinien. Diese Richtlinien sind einfacher zu verwalten.
- Mit ABAC ist es kein Problem, wenn Teams sich ändern bzw. schnell wachsen. Der Grund hierfür ist, dass Berechtigungen für neue Ressourcen automatisch basierend auf Attributen erteilt werden. Wenn Ihr Unternehmen beispielsweise bereits ABAC für die Projekte `Star` und `Heart` verwendet, ist es einfacher, ein neues `Lightning`-Projekt hinzuzufügen Ein IAM-Administrator erstellt eine neue Rolle mit dem `access-project` = `Lightning`-Tag. Es ist nicht notwendig, die Richtlinie zu ändern, um ein neues Projekt zu unterstützen. Jeder, der über Berechtigungen zur Übernahme der Rolle verfügt, kann Instances erstellen und anzeigen, die mit `access-project` = `Lightning` markiert sind. Darüber hinaus kann ein Teammitglied vom `Heart`-Projekt zum `Lightning`-Projekt wechseln. Der IAM-Administrator weist dem Benutzer zu einer anderen IAM-Rolle zu. Es ist nicht notwendig, die Berechtigungsrichtlinien zu ändern.
- Detaillierte Berechtigungen sind mit ABAC möglich. Wenn Sie Richtlinien erstellen, hat es sich bewährt, die geringsten Rechte zu erteilen. Mit herkömmlichem RBAC müssen Sie eine Richtlinie schreiben, die den Zugriff nur auf bestimmte Ressourcen erlaubt. Wenn Sie jedoch ABAC

verwenden, können Sie Aktionen für alle Ressourcen zulassen, aber nur, wenn das Ressourcen-Tag mit dem Tag des Prinzipals übereinstimmt.

- Verwenden Sie Mitarbeiterattribute aus Ihrem Firmenverzeichnis mit ABAC. Sie können Ihren SAML- oder OIDC-Anbieter so konfigurieren, dass er Sitzungs-Tags weitergibt. AWS Wenn sich Ihre Mitarbeiter zusammenschließen AWS, werden ihre Attribute auf den resultierenden Principal in angewendet. AWS Sie können dann ABAC verwenden, um Berechtigungen basierend auf diesen Attributen zuzulassen oder abzulehnen.

Ein ausführliches Tutorial, das die Verwendung von ABAC in demonstriert AWS, finden Sie unter. IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren

Sicherheitsfunktionen außerhalb von IAM

Sie verwenden IAM, um den Zugriff auf Aufgaben zu steuern, die mit den AWS Management ConsoleAWS Befehlszeilentools oder Service-API-Operationen mithilfe der AWS SDKs ausgeführt werden. Einige AWS Produkte bieten auch andere Möglichkeiten, ihre Ressourcen zu sichern. In der folgenden, nicht vollständigen Liste finden Sie einige Beispiele.

Amazon EC2

In Amazon Elastic Compute Cloud melden Sie sich mit einem Schlüsselpaar bei einer Instance (für Linux-Instances) bzw. mit einem Benutzernamen und einem Passwort (für Microsoft Windows-Instances) an.

Weitere Informationen finden Sie in der folgenden -Dokumentation:

- Erste Schritte mit Amazon EC2 EC2-Linux-Instances im Amazon EC2 EC2-Benutzerhandbuch
- Erste Schritte mit Amazon EC2 EC2-Windows-Instances im Amazon EC2 EC2-Benutzerhandbuch

Amazon RDS

In Amazon Relational Database Service melden Sie sich bei der Datenbank-Engine mit einem Benutzernamen und einem Passwort an, die beide an diese Datenbank gebunden sind.

Weitere Informationen finden Sie unter Erste Schritte mit Amazon RDS im Amazon RDS-Benutzerhandbuch.

Amazon EC2 und Amazon RDS

In Amazon EC2 und Amazon RDS steuern Sie mithilfe von Sicherheitsgruppen den Datenverkehr zu einer Instance oder Datenbank.

Weitere Informationen finden Sie in der folgenden -Dokumentation:

- Amazon EC2-Sicherheitsgruppen für Linux-Instances im Amazon EC2 EC2-Benutzerhandbuch
- Amazon EC2-Sicherheitsgruppen für Windows-Instances im Amazon EC2 EC2-Benutzerhandbuch
- Amazon EC2 Security Groups im Amazon RDS -Leitfaden

WorkSpaces

Bei Amazon melden WorkSpaces sich Benutzer mit einem Benutzernamen und einem Passwort auf einem Desktop an.

Weitere Informationen finden Sie unter Erste Schritte mit WorkSpaces im WorkSpaces Amazon-Administratorhandbuch.

Amazon WorkDocs

In Amazon erhalten Benutzer Zugriff auf geteilte Dokumente WorkDocs, indem sie sich mit einem Benutzernamen und einem Passwort anmelden.

Weitere Informationen finden Sie unter Erste Schritte mit Amazon WorkDocs im WorkDocs Amazon-Administratorhandbuch.

Diese Zugriffskontrollmethoden sind nicht Teil von IAM. Mit IAM können Sie steuern, wie diese AWS Produkte verwaltet werden — das Erstellen oder Beenden einer Amazon EC2 EC2-Instance, das Einrichten neuer Desktops usw. WorkSpaces Dies bedeutet, dass IAM Ihnen die Steuerung der ausgeführten Aufgaben erleichtert, indem Anforderungen an Amazon Web Services gestellt werden. IAM erleichtert auch die Steuerung des Zugriffs auf die AWS Management Console. IAM hilft Ihnen jedoch nicht dabei, die Sicherheit für Aufgaben wie die Anmeldung bei einem Betriebssystem (Amazon EC2), einer Datenbank (Amazon RDS), einem Desktop (Amazon) oder einer Website für die Zusammenarbeit (Amazon WorkSpaces WorkDocs) zu verwalten.

Wenn Sie mit einem bestimmten AWS Produkt arbeiten, lesen Sie unbedingt die Dokumentation, um mehr über die Sicherheitsoptionen für alle Ressourcen zu erfahren, die zu diesem Produkt gehören.

Quicklinks zu geläufigen Aufgaben

Verwenden Sie die folgenden Links, um Hilfe bei allgemeinen Aufgaben im Zusammenhang mit IAM zu erhalten.

Melden Sie sich für verschiedene Benutzertypen an

- Melden Sie sich bei der IAM-Konsole an, indem Sie IAM-Benutzer auswählen und Ihre AWS-Konto ID oder Ihren Kontoalias eingeben. Geben Sie auf der nächsten Seite Ihren IAM-Benutzernamen und Ihr Passwort ein.

  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 Benutzerhandbuch unter Anmeldung im AWSAWS-Anmeldung Access Portal.

  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.

  Unter Was ist AWS -Sign-In im AWS-Anmeldung -Benutzerhandbuch finden Sie Hilfe bei der Bestimmung Ihres Benutzertyps und der Anmeldeseite.

Verwalten von Passwörtern für -Benutzer

- Sie benötigen ein Passwort, um auf die zuzugreifen AWS Management Console, einschließlich des Zugriffs auf Rechnungsinformationen.

  Informationen dazu Root-Benutzer des AWS-Kontos finden Sie unter Ändern des Passworts für Root-Benutzer des AWS-Kontos im AWS Account Management Referenzhandbuch

  Für einen IAM-Benutzer finden Sie Informationen unter Verwalten von Passwörtern für IAM-Benutzer.

Verwalten von Berechtigungen für -Benutzer

- Sie verwenden Richtlinien, um den IAM-Benutzern in Ihrem AWS-Konto Bereich Berechtigungen zu gewähren. IAM-Benutzer haben keine Berechtigungen, wenn sie erstellt werden. Sie müssen also Berechtigungen hinzufügen, damit sie Ressourcen verwenden AWS können.

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in AWS IAM Identity Center:

  Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter Erstellen eines Berechtigungssatzes im AWS IAM Identity Center -Benutzerhandbuch.

- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

  Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter Erstellen einer Rolle für einen externen Identitätsanbieter (Verbund) im IAM-Benutzerhandbuch.

- IAM-Benutzer:
  - Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter Erstellen einer Rolle für einen IAM-Benutzer im IAM-Benutzerhandbuch.
  - (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter Hinzufügen von Berechtigungen zu einem Benutzer (Konsole) im IAM-Benutzerhandbuch.

Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien.

Listen Sie die Benutzer in Ihrem auf AWS-Konto und informieren Sie sich über deren Anmeldeinformationen

Siehe Abrufen von Berichten zu Anmeldeinformationen für Ihr AWS-Konto.

Hinzufügen der Multi-Factor Authentication (MFA)

Um ein virtuelles MFA-Gerät hinzuzufügen, lesen Sie einen der folgenden Abschnitte:

- Aktivieren eines virtuellen MFA-Geräts für Ihren Root-Benutzer des AWS-Kontos (Konsole)
- Aktivieren eines virtuellen MFA-Geräts für einen IAM-Benutzer (Konsole)

Um einen virtuellen FIDO-Sicherheitsschlüssel hinzuzufügen, lesen Sie einen der folgenden Abschnitte:

- Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für die Root-Benutzer des AWS-Kontos (Konsole)
- Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für einen anderen IAM-Benutzer (Konsole)

Um ein Hardware-MFA-Gerät hinzuzufügen, lesen Sie einen der folgenden Abschnitte:

- Aktivieren Sie ein Hardware-TOTP-Token für die Root-Benutzer des AWS-Kontos (Konsole).
- Aktivieren eines physischen TOTP-Tokens für einen anderen IAM-Benutzer (Konsole)

Abrufen eines Zugriffsschlüssels

Sie können einen Zugriffsschlüssel verwenden, um AWS Anfragen mithilfe der AWS SDKs, der AWS Befehlszeilentools oder der API-Operationen zu stellen.

> **Important**
>
> Als bewährte Methode empfiehlt es sich, temporäre Sicherheitsanmeldeinformationen (z. B. IAM-Rollen) zu verwenden, anstatt langfristige Anmeldeinformationen wie Zugriffsschlüssel zu erstellen. Bevor Sie Zugriffsschlüssel erstellen, prüfen Sie die Alternativen zu Langzeit-Zugriffsschlüsseln.

Anleitungen zum Schutz Ihrer Zugriffsschlüssel finden Sie unter Sichern von Zugriffsschlüsseln.

Informationen zum Verwalten von Zugriffsschlüsseln für einen IAM-Benutzer finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer.

Weitere Informationen zu den für Sie verfügbaren Sicherheitsanmeldedaten finden Sie AWS-Konto unter AWS Sicherheitsanmeldedaten.

Markieren von IAM-Ressourcen

Sie können die folgenden IAM-Ressourcen kennzeichnen:

- IAM-Benutzer
- IAM-Rollen
- Kundenverwaltete Richtlinien
- Identitätsanbieter
- Serverzertifikate
- Virtuelle MFA-Geräte

Informationen über Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.

Weitere Informationen zur Verwendung von Tags zur Steuerung des Zugriffs auf AWS Ressourcen finden Sie unterSteuern des Zugriffs auf AWS Ressourcen mithilfe von Tags.

Anzeigen der Aktionen, Ressourcen und Bedingungsschlüssel für alle Services

Diese Referenzdokumentation kann Ihnen bei der Erstellung detaillierter IAM-Richtlinien helfen. Jeder AWS -Service definiert die Aktionen, Ressourcen und Bedingungskontextschlüssel, die Sie

in IAM-Richtlinien verwenden. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Fangen Sie mit allen an AWS

Diese Dokumentation befasst sich in erster Linie mit dem IAM-Service. Informationen zu den ersten Schritten AWS und zur Nutzung mehrerer Dienste zur Lösung eines Problems, z. B. beim Erstellen und Starten Ihres ersten Projekts, finden Sie im Ressourcencenter für die ersten Schritte.

IAM-Konsolensuche

Verwenden Sie die Suchseite der IAM-Konsole als schnellere Option für die Suche nach IAM-Ressourcen. Sie können die Konsolensuche verwenden, um Zugriffsschlüssel zu finden, die sich auf Ihr Konto, IAM-Entitäten (wie Benutzer, Gruppen, Rollen, Identitätsanbieter), Richtlinien nach Namen und mehr beziehen.

Mit der Suchfunktion der IAM-Konsole können Sie Folgendes suchen:

- IAM-Entitätsnamen, die mit Ihren Suchbegriffen übereinstimmen (für Benutzer, Gruppen, Rollen, Identitätsanbieter und Richtlinien)
- Aufgaben, die mit Ihren Suchbegriffen übereinstimmen

Die IAM-Konsolen-Suchfunktion gibt keine Informationen über IAM Access Analyzer zurück.

Jede Zeile im Suchergebnis ist ein aktiver Link. Beispielsweise können Sie den Benutzernamen im Suchergebnis auswählen, mit dem Sie zur Detailseite dieses Benutzers gelangen. Sie können auch einen Link für eine Aktion auswählen, z. B. den Link für Benutzer erstellen, oder die Seite Create User (Benutzer erstellen) öffnen.

> **Note**
>
> Für die Suche nach Zugriffsschlüsseln müssen Sie die vollständige Zugriffsschlüssel-ID im Suchfeld eingeben. Im Suchergebnis wird der mit diesem Schlüssel verknüpfte Benutzer angezeigt. Von dort aus können Sie direkt zu der Seite dieses Benutzers navigieren, auf der Sie seinen Zugriffsschlüssel verwalten können.

Verwenden der IAM-Konsolensuche

Verwenden Sie die Seite Search in der IAM-Konsole, um die zu diesem Konto gehörenden Elemente zu suchen.

So suchen Sie nach Elementen in der IAM-Konsole

1. Folgen Sie dem Anmeldeverfahren, das Ihrem Benutzertyp entspricht, wie im Abschnitt So melden Sie sich bei AWS an im AWS -Anmelde-Benutzerhandbuch beschrieben.
2. Wählen Sie auf der Console Home (Konsolen-Startseite) den IAM-Service aus.
3. Klicken Sie im Navigationsbereich auf Suchen.
4. Geben Sie im Feld Search (Suchen) Ihre Suchbegriffe ein.
5. Wählen Sie in der Liste der Suchergebnisse einen Link aus, um zum entsprechenden Teil der Konsole zu navigieren.

Symbole in den Suchergebnissen der IAM-Konsole

Anhand der folgenden Symbole können Sie die Elementtypen identifizieren, die in einer Suche zurückgegeben werden:

Symbol	Beschreibung
	IAM-Benutzer
	IAM-Gruppen
	IAM-Rollen
	IAM-Richtlinien
	Aufgaben wie "Benutzer erstellen" oder "Richtlinie anfügen"

Symbol	Beschreibung
	Ergebnisse des Suchbegriffs **delete**

Beispiele für Suchausdrücke

Sie können die folgenden Ausdrücke bei der IAM-Suche verwenden. Ersetzen Sie die Begriffe in Kursivschrift durch die Namen der tatsächlichen IAM-Benutzer, -Gruppen, -Rollen, -Zugriffsschlüssel, -Richtlinien bzw. -Identitätsanbieter, nach denen Sie suchen möchten.

- ***user_name***, ***group_name*** , ***role_name***, ***policy_name*** oder ***identity_provider_name***
- ***access_key***
- **add user *user_name* to groups** oder **add users to group *group_name***
- **remove user *user_name* from groups**
- **delete *user_name*** oder **delete *group_name*** oder **delete *role_name*** oder **delete *policy_name*** oder **delete *identity_provider_name***
- **manage access keys *user_name***
- **manage signing certificates *user_name***
- **users**
- **manage MFA for *user_name***
- **manage password for *user_name***
- **create role**
- **password policy**
- **edit trust policy for role *role_name***
- **show policy document for role *role_name***
- **attach policy to *role_name***
- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***

- **attach entities to *policy_name***
- **detach entities from *policy_name***

AWS Identity and Access Management Ressourcen erstellen mit AWS CloudFormation

AWS Identity and Access Management ist in einen Service integriert AWS CloudFormation, der Ihnen hilft, Ihre AWS Ressourcen zu modellieren und einzurichten, sodass Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen müssen. Sie erstellen eine Vorlage, die alle gewünschten AWS Ressourcen beschreibt (z. B. Zugriffsschlüssel, Gruppen, Gruppenrichtlinien, Instanzprofile, verwaltete Richtlinien, OIDC-Anbieter, Inline-Richtlinien, Rollen, Rollenrichtlinien, SAML-Anbieter, Serverzertifikate, dienstbezogene Rollen, Benutzer (und Hinzufügen von Benutzern zu Gruppen), Benutzerrichtlinien und virtuelle MFA-Geräte) und diese Ressourcen für Sie AWS CloudFormation bereitstellt und konfiguriert.

Wenn Sie Ihre Vorlage verwenden AWS CloudFormation, können Sie sie wiederverwenden, um Ihre IAM-Ressourcen konsistent und wiederholt einzurichten. Beschreiben Sie Ihre Ressourcen einmal und stellen Sie dann dieselben Ressourcen immer wieder in mehreren Regionen AWS-Konten bereit.

IAM und Vorlagen AWS CloudFormation

Um Ressourcen für IAM und verwandte Dienste bereitzustellen und zu konfigurieren, müssen Sie sich mit Vorlagen auskennen AWS CloudFormation . Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren AWS CloudFormation Stacks bereitstellen möchten. Wenn Sie mit JSON oder YAML nicht vertraut sind, können Sie AWS CloudFormation Designer verwenden, um Ihnen die ersten Schritte mit Vorlagen zu erleichtern. AWS CloudFormation Weitere Informationen finden Sie unter Was ist AWS CloudFormation -Designer? im AWS CloudFormation -Benutzerhandbuch.

IAM unterstützt die Erstellung von Zugriffsschlüsseln, Gruppen, Gruppenrichtlinien, Instanzprofilen, verwalteten Richtlinien, OIDC-Anbietern, Inline-Richtlinien, Rollen, Rollenrichtlinien, SAML-Anbietern, Serverzertifikaten, serviceverknüpften Rollen, Benutzern (und Hinzufügen von Benutzern zu Gruppen), Benutzerrichtlinien und virtuellen MFA-Geräten in. AWS CloudFormationWeitere Informationen, einschließlich Beispielen für JSON- und YAML-Vorlagen für IAM-Ressourcen, finden Sie in der Referenz zum Ressourcentyp im AWS Identity and Access Management Benutzerhandbuch.AWS CloudFormation

Sie können auch Vorlagen erstellen, mit denen verwandte Ressourcen wie Rollen und verwaltete Richtlinien erstellt werden.

Erfahren Sie mehr über AWS CloudFormation

Weitere Informationen AWS CloudFormation dazu finden Sie in den folgenden Ressourcen:

- AWS CloudFormation
- AWS CloudFormation Benutzerhandbuch
- AWS CloudFormation API Reference
- AWS CloudFormation Benutzerhandbuch für die Befehlszeilenschnittstelle

Wird verwendet AWS CloudShell , um mit AWS Identity and Access Management zu arbeiten

AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt von der aus starten können. AWS Management Console Sie können AWS CLI Befehle für AWS Dienste (einschließlich AWS Identity and Access Management) mit Ihrer bevorzugten Shell (Bash PowerShell oder Z-Shell) ausführen. Und Sie können dies tun, ohne Befehlszeilentools herunterladen oder installieren zu müssen.

Sie starten AWS CloudShell von der aus AWS Management Console, und die AWS Anmeldeinformationen, mit denen Sie sich an der Konsole angemeldet haben, sind automatisch in einer neuen Shell-Sitzung verfügbar. Durch diese Vorauthentifizierung von AWS CloudShell Benutzern können Sie bei der Interaktion mit AWS Diensten wie IAM mithilfe von AWS CLI Version 2 (vorinstalliert in der Rechenumgebung der Shell) die Konfiguration der Anmeldeinformationen überspringen.

Erhalt von IAM-Berechtigungen für AWS CloudShell

Mithilfe der von bereitgestellten Ressourcen zur Zugriffsverwaltung können Administratoren IAM-Benutzern Berechtigungen erteilen AWS Identity and Access Management, sodass sie auf die Funktionen der Umgebung zugreifen AWS CloudShell und diese nutzen können.

Am schnellsten kann ein Administrator Benutzern Zugriff gewähren, indem er eine AWS verwaltete Richtlinie verwendet. Bei einer von AWS verwalteten Richtlinie handelt es sich um eine eigenständige

Richtlinie, die von AWS erstellt und verwaltet wird. Die folgende AWS verwaltete Richtlinie für CloudShell kann an IAM-Identitäten angehängt werden:

- `AWSCloudShellFullAccess`: Erteilt die Erlaubnis zur Nutzung AWS CloudShell mit vollem Zugriff auf alle Funktionen.

Wenn Sie den Umfang der Aktionen einschränken möchten, die ein IAM-Benutzer ausführen kann AWS CloudShell, können Sie eine benutzerdefinierte Richtlinie erstellen, die die `AWSCloudShellFullAccess` verwaltete Richtlinie als Vorlage verwendet. Weitere Informationen zur Einschränkung der Aktionen, die Benutzern zur Verfügung stehen CloudShell, finden Sie im AWS CloudShell Benutzerhandbuch unter Verwaltung von AWS CloudShell Zugriff und Nutzung mit IAM-Richtlinien.

Interaktion mit IAM mithilfe von AWS CloudShell

Nach dem Start AWS CloudShell von der AWS Management Console können Sie sofort mit der Interaktion mit IAM über die Befehlszeilenschnittstelle beginnen.

> **Note**
>
> Wenn Sie AWS CLI in verwenden AWS CloudShell, müssen Sie keine zusätzlichen Ressourcen herunterladen oder installieren. Da Sie außerdem bereits in der Shell authentifiziert sind, müssen Sie vor dem Tätigen von Anrufen keine Anmeldeinformationen konfigurieren.

Erstellen Sie eine IAM-Gruppe und fügen Sie der Gruppe einen IAM-Benutzer hinzu mit AWS CloudShell

Im folgenden Beispiel wird CloudShell eine IAM-Gruppe erstellt, der Gruppe ein IAM-Benutzer hinzugefügt und anschließend überprüft, ob der Befehl erfolgreich ausgeführt wurde.

1. Von der aus können Sie starten AWS Management Console, CloudShell indem Sie die folgenden Optionen auswählen, die in der Navigationsleiste verfügbar sind:

   - Wählen Sie das CloudShell Symbol.
   - Geben Sie „Cloudshell“ in das Suchfeld ein und wählen Sie dann die CloudShell Option.

2. Um eine IAM-Gruppe zu erstellen, geben Sie den folgenden Befehl in die CloudShell Befehlszeile ein. In diesem Beispiel haben wir die Gruppe east_coast: genannt:

```
aws iam create-group --group-name east_coast
```

Wenn der Aufruf erfolgreich ist, zeigt die Befehlszeile eine Antwort des Services an, die der folgenden Ausgabe ähnelt:

```
{
   "Group": {
       "Path": "/",
       "GroupName": "east_coast",
       "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
       "Arn": "arn:aws:iam::111122223333:group/east_coast",
       "CreateDate": "2023-09-11T21:02:21+00:00"
    }
}
```

3. Um der von Ihnen erstellten Gruppe einen Benutzer hinzuzufügen, verwenden Sie den folgenden Befehl und geben Sie den Gruppennamen und den Benutzernamen an. In diesem Beispiel haben wir die Gruppe east_coast und den Benutzer johndoe benannt:

```
aws iam add-user-to-group --group-name east_coast --user-name johndoe
```

4. Um zu überprüfen, ob der Benutzer in der Gruppe ist, verwenden Sie den folgenden Befehl und geben Sie dabei den Gruppennamen an. In diesem Beispiel verwenden wir weiterhin die Gruppe east_coast:

```
aws iam get-group --group-name east_coast
```

Wenn der Aufruf erfolgreich ist, zeigt die Befehlszeile eine Antwort des Services an, die der folgenden Ausgabe ähnelt:

```
{
  "Users": [
    {
        "Path": "/",
```

```
            "UserName": "johndoe",
            "UserId": "AIDAYBDBW4JBXGEXAMPLE",
            "Arn": "arn:aws:iam::552108220995:user/johndoe",
            "CreateDate": "2023-09-11T20:43:14+00:00",
            "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
        }
    ],
    "Group": {
            "Path": "/",
            "GroupName": "east_coast",
            "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
            "Arn": "arn:aws:iam::111122223333:group/east_coast",
            "CreateDate": "2023-09-11T21:02:21+00:00"
        }
}
```

Verwenden von IAM mit einem SDK AWS

AWS Software Development Kits (SDKs) sind für viele gängige Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK for C++	AWS SDK for C++ Code-Beispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK for Go	AWS SDK for Go Code-Beispiele
AWS SDK for Java	AWS SDK for Java Code-Beispiele
AWS SDK for JavaScript	AWS SDK for JavaScript Code-Beispiele
AWS SDK for Kotlin	AWS SDK for Kotlin Code-Beispiele
AWS SDK for .NET	AWS SDK for .NET Code-Beispiele
AWS SDK for PHP	AWS SDK for PHP Code-Beispiele

SDK-Dokumentation	Codebeispiele
AWS Tools for PowerShell	Tools für PowerShell Codebeispiele
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) Code-Beispiele
AWS SDK for Ruby	AWS SDK for Ruby Code-Beispiele
AWS SDK for Rust	AWS SDK for Rust Code-Beispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Code-Beispiele
AWS SDK for Swift	AWS SDK for Swift Code-Beispiele

Beispiele, die sich speziell auf IAM beziehen, finden Sie unter Codebeispiele für IAM mit SDKs AWS.

> **Beispiel für die Verfügbarkeit**
>
> Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

Einrichten von IAM

> **Important**
>
> Bewährte Methoden für IAM empfehlen, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um mit temporären Anmeldeinformationen zugreifen zu AWS können, anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen zu verwenden.

AWS Identity and Access Management (IAM) hilft Ihnen dabei, den Zugriff auf Amazon Web Services (AWS) und Ihre Kontoressourcen sicher zu kontrollieren. IAM kann auch Ihre Anmeldeinformationen geheim halten. Sie müssen sich nicht registrieren, um IAM zu verwenden. Die Nutzung von IAM ist kostenlos.

Verwenden Sie IAM, um Identitäten wie Benutzern und Rollen Zugriff auf Ressourcen in Ihrem Konto zu geben. Sie können IAM beispielsweise mit bestehenden Benutzern in Ihrem Unternehmensverzeichnis verwenden, die Sie extern verwalten, AWS oder Sie können Benutzer erstellen, die Sie verwenden. AWS AWS IAM Identity Center Verbundidentitäten übernehmen definierte IAM-Rollen, um auf die Ressourcen zuzugreifen, die sie benötigen. Weitere Informationen zu IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

> **Note**
>
> IAM ist in mehrere AWS Produkte integriert. Eine Liste der Services, die IAM unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

Themen

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.

Auf die geringsten Berechtigungen vorbereiten

Die Verwendung von geringsten Berechtigungen ist eine bewährte IAM-Empfehlung. Das Konzept der geringsten Berechtigungen besteht darin, den Benutzern ausschließlich die Berechtigungen zu

erteilen, die zum Ausführen einer Aufgabe erforderlich sind. Überlegen Sie sich bei der Einrichtung, wie Sie geringste Berechtigungen unterstützen werden. Sowohl der Root-Benutzer als auch der Administrator-Benutzer verfügen über umfangreiche Berechtigungen, die für alltägliche Aufgaben nicht erforderlich sind. Während Sie sich mit verschiedenen Diensten AWS vertraut machen und diese testen, empfehlen wir Ihnen, mindestens einen zusätzlichen Benutzer in IAM Identity Center mit geringeren Berechtigungen zu erstellen, den Sie in verschiedenen Szenarien verwenden können. Sie können IAM-Richtlinien verwenden, um die Aktionen zu definieren, die unter bestimmten Bedingungen auf bestimmten Ressourcen ausgeführt werden können, und dann mit Ihrem Konto mit weniger Berechtigungen eine Verbindung mit diesen Ressourcen herzustellen.

Wenn Sie IAM Identity Center verwenden, sollten Sie zunächst die Verwendung von IAM-Identity-Center-Berechtigungssätzen in Betracht ziehen. Weitere Informationen finden Sie unter Erstellen eines Berechtigungssatzes im IAM-Identity-Center-Benutzerhandbuch.

Wenn Sie IAM Identity Center nicht verwenden, definieren Sie die Berechtigungen für verschiedene IAM-Entitäten mithilfe von IAM-Rollen. Weitere Informationen hierzu finden Sie unter Erstellen von IAM-Rollen.

Sowohl IAM-Rollen als auch IAM Identity Center-Berechtigungssätze können AWS verwaltete Richtlinien verwenden, die auf Aufgabenfunktionen basieren. Einzelheiten zu den durch diese Richtlinien gewährten Berechtigungen finden Sie unter AWS verwaltete Richtlinien für Jobfunktionen.

> **Important**
>
> Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie von allen Kunden verwendet werden können. AWS Wir empfehlen die geringsten Berechtigungen mit IAM Access Analyzer zu verwenden, um die geringsten Berechtigungen auf der Grundlage Ihrer in AWS CloudTrail protokollierten Zugriffsaktivitäten zu erstellen. Weitere Informationen zur Richtlinienerstellung finden Sie unter IAM Access Analyzer Richtlinienerstellung.

IAM-Verwaltungsmethoden

Sie können IAM entweder über die AWS Konsole, die AWS Befehlszeilenschnittstelle oder über die Anwendungsschnittstellen (APIs) in den zugehörigen SDKs verwalten. Überlegen Sie beim

Einrichten, welche Methoden Sie unterstützen möchten und wie Sie die verschiedenen Benutzer unterstützen möchten.

Themen

- AWS Konsole
- AWS Befehlszeilenschnittstelle (CLI) und Software Development Kits (SDKs)

AWS Konsole

Die AWS Management Console ist eine Webanwendung, die eine breite Sammlung von Servicekonsolen für die Verwaltung von AWS Ressourcen umfasst und sich auf diese bezieht. Wenn Sie sich zum ersten Mal anmelden, sehen Sie die Startseite der Konsole. Die Startseite bietet Zugriff auf jede Servicekonsole und bietet einen zentralen Ort, an dem Sie auf die Informationen zugreifen können, die Sie für die Ausführung der AWS damit verbundenen Aufgaben benötigen. Welche Dienste und Anwendungen Ihnen nach der Anmeldung an der Konsole zur Verfügung stehen, hängt davon ab, auf welche AWS Ressourcen Sie zugreifen dürfen. Sie können Berechtigungen für Ressourcen erhalten, indem Sie entweder eine Rolle übernehmen, Mitglied einer Gruppe sind, der Berechtigungen gewährt wurden, oder indem Ihnen explizit eine Berechtigung gewährt wird. Bei einem eigenständigen AWS Konto konfiguriert der Root-Benutzer oder IAM-Administrator den Zugriff auf Ressourcen. Denn AWS Organizations das Verwaltungskonto oder der delegierte Administrator konfiguriert den Zugriff auf Ressourcen.

Wenn Sie planen, dass Benutzer die AWS Management Console zur Verwaltung von AWS Ressourcen verwenden, empfehlen wir aus Sicherheitsgründen, Benutzer mit temporären Anmeldeinformationen zu konfigurieren. IAM-Benutzer, die eine Rolle übernommen haben, Verbundbenutzer und Benutzer im IAM Identity Center verfügen über temporäre Anmeldeinformationen, während der IAM-Benutzer und der Root-Benutzer über langfristige Anmeldeinformationen verfügen. Anmeldeinformationen für Root-Benutzer bieten vollständigen Zugriff auf das AWS-Konto, während andere Benutzer über Anmeldeinformationen verfügen, die diesen Zugriff auf die Ressourcen gewähren, die ihnen durch IAM-Richtlinien gewährt werden.

Das Anmeldeerlebnis ist für die verschiedenen AWS Management Console Benutzertypen unterschiedlich.

- IAM-Benutzer und Root-Benutzer melden sich über die Haupt-Anmelde-URL an AWS (https://signin.aws.amazon.com). Sobald sie diese angemeldet haben, haben sie Zugriff auf die Ressourcen in dem Konto, für das ihnen die Berechtigung gewährt wurde.

Um sich als Root-Benutzer anzumelden, benötigen Sie die E-Mail-Adresse und das Passwort für den Root-Benutzer.

Um sich als IAM-Benutzer anzumelden, benötigen Sie die AWS-Konto Nummer oder den Alias, den IAM-Benutzernamen und das IAM-Benutzerkennwort.

Wir empfehlen Ihnen, IAM-Benutzer in Ihrem Konto auf bestimmte Situationen zu beschränken, die langfristige Anmeldeinformationen erfordern, z. B. für den Zugriff in Notfällen, und den Root-Benutzer nur für Aufgaben zu verwenden, die Anmeldeinformationen des Root-Benutzers erfordern.

Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich den IAM-Benutzernamen und die Kontoinformationen zu merken. Wenn der Benutzer das nächste Mal eine Seite in der aufruft AWS Management Console, verwendet die Konsole das Cookie, um den Benutzer auf die Anmeldeseite für das Konto weiterzuleiten.

Melden Sie sich nach Abschluss Ihrer Sitzung von der Konsole ab, um eine Wiederverwendung Ihrer vorherigen Anmeldung zu verhindern.

- IAM Identity Center-Benutzer melden sich über ein bestimmtes AWS Zugriffsportal an, das nur für ihre Organisation gilt. Sobald sie sich angemeldet haben, können sie auswählen, auf welches Konto oder welche Anwendung sie zugreifen möchten. Wenn diese auf ein Konto zugreifen möchten, wählen sie aus, welchen Berechtigungssatz sie für die Verwaltungssitzung verwenden möchten.
- Verbundbenutzer, die bei einem externen Identitätsanbieter verwaltet werden und mit der Anmeldung bei einem AWS-Konto über ein benutzerdefiniertes Unternehmenszugriffsportal verknüpft sind. Die AWS Ressourcen, die Verbundbenutzern zur Verfügung stehen, hängen von den Richtlinien ab, die von ihrer Organisation ausgewählt wurden.

> **Note**
>
> Um ein zusätzliches Maß an Sicherheit zu bieten, können Root-Benutzer, IAM-Benutzer und Benutzer in IAM Identity Center die Multi-Faktor-Authentifizierung (MFA) verifizieren lassen, AWS bevor sie Zugriff auf Ressourcen gewähren. AWS Wenn MFA aktiviert ist, müssen Sie auch über Zugriff auf das MFA-Gerät verfügen, um sich anzumelden.

Weitere Informationen darüber, wie sich verschiedene Benutzer an der Management-Konsole anmelden, finden Sie im Anmelde-Benutzerhandbuch unter Anmeldung an der AWS Management Console.AWS

AWS Befehlszeilenschnittstelle (CLI) und Software Development Kits (SDKs)

IAM-Identity-Center- und IAM-Benutzer verwenden unterschiedliche Methoden zur Authentifizierung ihrer Anmeldeinformationen, wenn sie sich über die CLI oder die Anwendungsschnittstellen (APIs) in den zugehörigen SDKs authentifizieren.

Anmeldeinformationen und Konfigurationseinstellungen befinden sich an mehreren Stellen, z. B. in den System- oder Benutzerumgebungsvariablen, in lokalen AWS Konfigurationsdateien oder werden explizit in der Befehlszeile als Parameter deklariert. Bestimmte Speicherorte haben Vorrang vor anderen.

Sowohl IAM Identity Center als auch IAM stellen Zugriffsschlüssel bereit, die mit der CLI oder dem SDK verwendet werden können. Bei Zugriffsschlüsseln des IAM Identity Center handelt es sich um temporäre Anmeldeinformationen, die automatisch aktualisiert werden können. Sie werden im Vergleich zu Langzeit-Zugriffsschlüsseln für IAM-Benutzer empfohlen.

Um Ihre AWS-Konto Nutzung der CLI oder des SDK zu verwalten, können Sie sie AWS CloudShell von Ihrem Browser aus verwenden. Wenn Sie CloudShell CLI- oder SDK-Befehle ausführen, müssen Sie sich zuerst bei der Konsole anmelden. Die Berechtigungen für den Zugriff auf AWS Ressourcen basieren auf den Anmeldeinformationen, mit denen Sie sich an der Konsole angemeldet haben. Abhängig von Ihrer Erfahrung ist die CLI möglicherweise eine effizientere Methode zur Verwaltung Ihres AWS-Konto.

Für die Anwendungsentwicklung können Sie die CLI oder das SDK auf Ihren Computer herunterladen und sich über die Eingabeaufforderung oder ein Docker-Fenster anmelden. In diesem Szenario konfigurieren Sie Authentifizierung und Zugangsdaten als Teil des CLI-Skripts oder der SDK-Anwendung. Sie können den programmgesteuerten Zugriff auf Ressourcen je nach Umgebung und verfügbarem Zugriff auf unterschiedliche Weise konfigurieren.

- Zu den empfohlenen Optionen für die Authentifizierung von lokalem Code mit dem AWS Dienst gehören IAM Identity Center und IAM Roles Anywhere
- Zu den empfohlenen Optionen für die Authentifizierung von Code, der in einer AWS Umgebung ausgeführt wird, gehören die Verwendung von IAM-Rollen oder die Verwendung von IAM Identity Center-Anmeldeinformationen.

Wenn Sie IAM Identity Center verwenden, können Sie kurzfristige Anmeldeinformationen auf der Startseite des AWS -Zugriffsportals abrufen, wo Sie Ihren Berechtigungssatz auswählen. Diese Anmeldeinformationen haben eine definierte Dauer und werden nicht automatisch aktualisiert. Wenn Sie diese Anmeldeinformationen verwenden möchten, wählen Sie nach der Anmeldung im AWS Portal den AWS-Konto und anschließend den Berechtigungssatz aus. Wählen Sie Befehlszeilen- oder programmatischer Zugriff aus, um die Optionen anzuzeigen, mit denen Sie programmgesteuert oder über die CLI auf AWS Ressourcen zugreifen können. Weitere Informationen zu diesen Methoden finden Sie unter Abrufen und Aktualisieren temporärer Anmeldeinformationen im IAM-Identity-Center-Benutzerhandbuch. Diese Anmeldeinformationen werden häufig während der Anwendungsentwicklung verwendet, um Code schnell zu testen.

Wir empfehlen die Verwendung von IAM Identity Center-Anmeldeinformationen, die bei der Automatisierung des Zugriffs auf Ihre Ressourcen automatisch aktualisiert werden. AWS Wenn Sie Benutzer und Berechtigungssätze in IAM Identity Center konfiguriert haben, verwenden Sie den `aws configure sso`-Befehl, um einen Befehlszeilenassistenten zu verwenden, der Ihnen hilft, die für Sie verfügbaren Anmeldeinformationen zu identifizieren und sie in einem Profil zu speichern. Weitere Informationen zur Konfiguration Ihres Profils finden Sie unter Konfiguration Ihres Profils mit dem `aws configure sso`-Assistenten im Benutzerhandbuch für die AWS -Befehlszeilenschnittstelle für Version 2.

> **Note**
>
> Viele Beispielanwendungen verwenden Langzeit-Zugriffsschlüsseln, die IAM-Benutzern oder Root-Benutzern zugeordnet sind. Im Rahmen einer Lernübung sollten Sie langfristige Anmeldeinformationen nur in einer Sandbox-Umgebung verwenden. Informieren Sie sich über die Alternativen zu Langzeit-Zugriffsschlüsseln und planen Sie, Ihren Code so bald wie möglich auf alternative Anmeldeinformationen wie IAM-Identity-Center-Anmeldeinformationen oder IAM-Rollen umzustellen. Löschen Sie nach der Umstellung Ihres Codes die Zugriffsschlüssel.

Weitere Informationen zur Konfiguration der CLI finden Sie unter Installieren oder Aktualisieren der neuesten Version der AWS CLI im AWS Command Line Interface User Guide für Version 2 und Authentifizierungs- und Zugriffsanmeldeinformationen im AWS Command Line Interface User Guide

Weitere Informationen zur Konfiguration des SDK finden Sie unter IAM-Identity-Center-Authentifizierung im Referenzhandbuch für AWS -SDKs und Tools und unter IAM Roles Anywhere im AWS -SDKs- und Tools-Referenzhandbuch.

Deine AWS-Konto ID und ihr Alias

IAM-Benutzer im Konto melden sich mit einer Web-URL an, die entweder den Konto-Alias oder eine Konto-ID enthält. Wenn Sie die URL nicht haben, müssen Sie auf der AWS Anmeldeseite entweder den AWS-Konto Alias oder die Konto-ID angeben.

Falls Sie Ihre Konto-ID oder Ihren Alias nicht kennen, können Sie wie folgt vorgehen:

- Überprüfen Sie Ihren Browserverlauf. Wenn Sie sich schon einmal angemeldet haben, ist die Information möglicherweise in Ihren kürzlich besuchten Websites gespeichert.
- Wenn Sie die AWS CLI oder ein AWS SDK mit Ihren Kontoanmeldeinformationen konfiguriert haben, können Sie Ihre Konto-ID aus Ihren Konfigurationsdateien abrufen.
- Fragen Sie Ihren lokalen Administrator oder Kontoinhaber. Konto-IDs AWS können Benutzern nicht zur Verfügung gestellt werden.

> **Tip**
>
> Um ein Lesezeichen für die Anmeldeseite Ihres Kontos in Ihrem Webbrowser zu erstellen, sollten Sie die URL der Anmeldeseite Ihres Kontos manuell als Lesezeichen eintragen. Verwenden Sie nicht das Lesezeichen-Feature Ihres Webbrowsers, da dadurch spezifische Informationen im Zusammenhang mit Ihrer aktuellen Browsersitzung erfasst werden, die zukünftige Besuche der Anmeldeseite beeinträchtigen.

Themen

Zeigen Sie Ihre AWS-Konto ID an

Sie können die Konto-ID für Sie AWS-Konto mithilfe der folgenden Methoden einsehen.

Anzeigen Ihrer Konto-ID mithilfe der Konsole

Die Konto-ID wird auf dem IAM-Dashboard im AWS-Konto Abschnitt angezeigt. Abhängig von Ihrem Benutzertyp gibt es weitere Möglichkeiten, Ihre Konto-ID in der Konsole anzuzeigen. Wenn Sie eine Rolle übernommen haben, sind Sicherheitsanmeldedaten nicht verfügbar.

Benutzertyp	Verfahren
Stammbenutzer	Wählen Sie in der Navigationsleiste oben rechts Ihren Benutzernamen und dann Sicherheitsanmeldedaten aus. Die Kontonummer wird unter Kontokennungen angezeigt.
IAM-Benutzer	Wählen Sie in der Navigationsleiste oben rechts Ihren Benutzernamen aus. Die Konto-ID wird über Ihrem Benutzernamen angezeigt. Wählen Sie Sicherheitsanmeldeinformationen aus. Die Kontonummer wird unter Kontodetails angezeigt.
Verbundbenutzer	Wählen Sie in der Navigationsleiste oben rechts Ihren Benutzernamen aus. Die Konto-ID wird über Ihrem Benutzernamen angezeigt.
Angenommene Rolle	Wählen Sie in der Navigationsleiste oben rechts das Support-Symbol und dann Support Center aus der Liste aus. Ihre aktuell angemeldete 12-stellige Kontonummer (ID) wird im Bereich Support-Center-Navigationsbereich.

Rufen Sie Ihre Konto-ID auf, indem Sie AWS CLI

Verwenden Sie den folgenden Befehl, um Ihre Benutzer-ID, Konto-ID und Ihren Benutzer-ARN anzuzeigen:

- aws sts get-caller-identity

Anzeigen Ihrer Konto-ID mithilfe der API

Verwenden Sie die folgende API, um Ihre Benutzer-ID, Konto-ID und Ihren Benutzer-ARN anzuzeigen:

- GetCallerIdentity

Über Konto-Aliasse

Wenn Sie möchten, dass die URL für Ihre Anmeldeseite Ihren Firmennamen (oder eine andere benutzerfreundliche Kennung) anstelle Ihrer AWS-Konto ID enthält, können Sie einen Kontoalias erstellen. Dieser Abschnitt enthält Informationen über AWS-Konto Aliase und listet die API-Operationen auf, mit denen Sie einen Alias erstellen.

Die URL Ihrer Anmeldeseite hat standardmäßig das folgende Format.

```
https://Your_Account_ID.signin.aws.amazon.com/console/
```

Wenn Sie einen AWS-Konto Alias für Ihre AWS-Konto ID erstellen, sieht die URL Ihrer Anmeldeseite wie im folgenden Beispiel aus.

```
https://Your_Account_Alias.signin.aws.amazon.com/console/
```

Überlegungen

- Sie AWS-Konto können nur einen Alias haben. Wenn Sie einen neuen Alias für ein AWS -Konto erstellen, wird der vorherige Alias vom neuen Alias überschrieben und die URL mit dem vorherigen Alias wird funktionsunfähig.
- Der Kontoalias darf nur Zahlen, Kleinbuchstaben und Bindestriche enthalten. Weitere Informationen zu Einschränkungen für AWS Kontoentitäten finden Sie unterIAM und Kontingente AWS STS.
- Der Kontoalias muss für alle Amazon-Web-Services-Produkte innerhalb einer bestimmten Netzwerkpartition eindeutig sein.

  Eine Partition ist eine Gruppe von AWS Regionen. Jedes AWS -Konto ist auf eine Partition angelegt.

  Im Folgenden werden die unterstützten Partitionen angezeigt:

- `aws`- AWS Regionen
- `aws-cn` – Chinesische Regionen
- `aws-us-gov`- AWS GovCloud (US) Regionen

Erstellen, Löschen und Auflisten eines AWS-Konto -Alias

Sie können die AWS Management Console, die IAM-API oder die Befehlszeilenschnittstelle verwenden, um Ihren AWS-Konto Alias zu erstellen oder zu löschen.

> **Note**
>
> Kontoaliase sind keine Geheimnisse und werden auf der URL Ihrer öffentlichen Anmeldeseite angezeigt. Nehmen Sie keine vertraulichen Informationen in den Alias Ihres Kontos auf. Die ursprüngliche URL mit Ihrer AWS-Konto ID bleibt aktiv und kann verwendet werden, nachdem Sie Ihren AWS-Konto Alias erstellt haben.

Erstellen oder Bearbeiten eines Konto-Alias (Konsole)

Sie können einen Kontoalias mit der AWS Management Console erstellen, bearbeiten und löschen.

> **Mindestberechtigungen**
>
> Für die folgenden Schritte sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`

So erstellen oder bearbeiten Sie einen Konto-Alias (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. Wählen Sie im Navigationsbereich Dashboard (Dashboard).
3. Wählen Sie im Abschnitt AWS -Konto neben Konto-Alias die Option Erstellen aus. Wenn bereits ein Alias existiert, wählen Sie Bearbeiten.

4. Geben Sie im Dialogfeld den gewünschten Namen für den Alias ein und wählen Sie Änderungen speichern aus.

> **Note**
>
> Ihnen kann jeweils nur ein Alias zugeordnet sein. AWS-Konto Wenn Sie einen neuen Alias erstellen, wird der vorherige Alias entfernt und die Anmelde-URL, die mit dem vorherigen Alias verknüpft war, funktioniert nicht mehr.

Löschen eines Konto-Alias (Konsole)

Sie können einen Konto-Alias über die AWS Management Console löschen.

> **Mindestberechtigungen**
>
> Für die folgenden Schritte sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

So löschen Sie einen Konto-Alias (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. Wählen Sie im Navigationsbereich Dashboard (Dashboard).
3. Wählen Sie im Abschnitt AWS -Konto neben Konto-Alias die Option Löschen aus.

> **Note**
>
> Die einzige Anmelde-URL für Ihr Konto basiert auf Ihrer Konto-ID. Versuche, eine Verbindung mit der Alias-URL herzustellen, werden nicht umgeleitet.

Erstellen, Löschen und Auflisten von Aliasnamen (AWS CLI)

> **Note**
>
> Für die Verwendung der folgenden Befehle sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Führen Sie den folgenden Befehl aus, um einen Alias für die URL Ihrer AWS Management Console Anmeldeseite zu erstellen:

- `aws iam create-account-alias`

Führen Sie den folgenden Befehl aus, um einen AWS-Konto ID-Alias zu löschen:

- `aws iam delete-account-alias`

Führen Sie den folgenden Befehl aus, um Ihren AWS-Konto ID-Alias anzuzeigen:

- `aws iam list-account-aliases`

Example Alias-Befehle

Führen Sie den folgenden Befehl aus, um Ihren AWS-Konto ID-Alias anzuzeigen.

```
$ aws iam list-account-aliases
{
    "AccountAliases": [
        "myaccountalias"
    ]
}
```

Führen Sie den folgenden Befehl aus, um einen Alias für Ihre AWS Management Console Anmeldung zu erstellen:

```
$ aws iam create-account-alias \
    --account-alias myaliasname
```

Dieser Befehl erzeugt keine Ausgabe, wenn er erfolgreich ist.

Führen Sie den folgenden Befehl aus, um einen AWS-Konto ID-Alias zu löschen.

```
$ aws iam delete-account-alias \
--account-alias myaliasname
```

Dieser Befehl erzeugt keine Ausgabe, wenn er erfolgreich ist.

Erstellen, Löschen und Auflisten von Aliasnamen (AWS -API)

> **Note**
>
> Für die Verwendung der folgenden API-Vorgänge sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Rufen Sie den folgenden Vorgang auf, um einen Alias für die URL Ihrer AWS Management Console Anmeldeseite zu erstellen:

- `CreateAccountAlias`

Rufen Sie den folgenden Vorgang auf, um einen AWS-Konto ID-Alias zu löschen:

- `DeleteAccountAlias`

Rufen Sie den folgenden Vorgang auf, um Ihren AWS-Konto ID-Alias anzuzeigen:

- `ListAccountAliases`

Erste Schritte mit IAM

Verwenden Sie dieses Tutorial, um mit AWS Identity and Access Management (IAM) zu beginnen. Sie erfahren, wie Sie Rollen, Benutzer und Richtlinien mithilfe der AWS Management Console erstellen.

AWS Identity and Access Management ist eine Funktion, die Sie ohne zusätzliche Kosten AWS-Konto anbieten. Ihnen wird nur die Nutzung anderer AWS Produkte durch Ihre IAM-Benutzer in Rechnung gestellt. Informationen zu den Preisen anderer AWS Produkte finden Sie auf der Preisseite von Amazon Web Services.

> **Note**
>
> Diese Dokumentation befasst sich in erster Linie mit dem IAM-Service. Informationen zu den ersten Schritten AWS und zur Nutzung mehrerer Dienste zur Lösung eines Problems, z. B. beim Aufbau und Start Ihres ersten Projekts, finden Sie im Ressourcencenter für die ersten Schritte.

Inhalt

Voraussetzungen

Bevor Sie beginnen, sollten Sie sicherstellen, dass Sie die in Einrichten von IAM beschriebenen Schritte ausgeführt haben. In diesem Tutorial wird das Administratorkonto verwendet, das Sie in diesem Verfahren erstellt haben.

Erstellen Ihres ersten IAM-Benutzers

Ein IAM-Benutzer ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Benutzer können in Gruppen organisiert werden, die dieselben Berechtigungen haben.

> **Note**
>
> Im Sinne bewährter Sicherheitsmethoden wird empfohlen, den Zugriff auf Ihre Ressourcen über einen Identitätsverbund zu ermöglichen, anstatt IAM-Benutzer zu erstellen. Informationen zu bestimmten Situationen, in denen ein IAM-Benutzer erforderlich ist, finden Sie unter Wann sollte ein IAM-Benutzer (anstelle einer Rolle) erstellt werden?.

Um Sie mit dem Prozess der Erstellung eines IAM-Benutzers vertraut zu machen, werden Sie in diesem Tutorial Schritt für Schritt durch die Erstellung eines IAM-Benutzers und einer IAM-Gruppe für den Notfallzugriff geführt.

So erstellen Sie Ihren ersten IAM-Benutzer

1. Folgen Sie dem Anmeldeverfahren, das Ihrem Benutzertyp entspricht, wie im Abschnitt So melden Sie sich bei AWS an im AWS -Anmelde-Benutzerhandbuch beschrieben.
2. Wählen Sie auf der Console Home (Konsolen-Startseite) den IAM-Service aus.
3. Wählen Sie im Navigationsbereich Users (Benutzer) und dann Add users (Benutzer hinzufügen) aus.

   > **Note**
   >
   > Wenn Sie IAM Identity Center aktiviert haben, AWS Management Console wird eine Erinnerung angezeigt, dass es am besten ist, den Benutzerzugriff in IAM Identity Center zu verwalten. In diesem Tutorial wird der von Ihnen erstellte IAM-Benutzer nur verwendet, wenn die Anmeldeinformationen Ihres Benutzers in IAM Identity Center nicht verfügbar sind.

4. Geben Sie unter User Name (Benutzername) den Text **`EmergencyAccess`** ein. Namen dürfen keine Leerzeichen enthalten.
5. Aktivieren Sie das Kontrollkästchen neben Benutzerzugriff auf bereitstellen AWS Management Console— optional und wählen Sie dann Ich möchte einen IAM-Benutzer erstellen aus.

6. Wählen Sie unter Console password (Konsolenpasswort) Autogenerated password (Automatisch generiertes Passwort).
7. Deaktivieren Sie das Kontrollkästchen neben User must create a new password at next sign-in (recommended) (Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen). Da dieser IAM-Benutzer für den Notfallzugriff vorgesehen ist, behält ein vertrauenswürdiger Administrator das Passwort und gibt es nur bei Bedarf weiter.
8. Wählen Sie auf der Seite Set permissions (Berechtigungen festlegen) unter Permissions options (Berechtigungsoptionen) die Option Add user to group (Benutzer zur Gruppe hinzufügen) aus. Wählen Sie dann unter User groups (Benutzergruppen) die Option Create group (Gruppe erstellen) aus.
9. Geben Sie auf der Seite Create user group (Benutzergruppe erstellen) im Feld User group name (Benutzergruppenname) **`EmergencyAccessGroup`** ein. Wählen Sie dann unter Berechtigungsrichtlinien die Option aus AdministratorAccess.
10. Wählen Sie Create user group (Benutzergruppe erstellen) aus, um zur Seite Set permissions (Berechtigungen festlegen) zurückzukehren.
11. Wählen Sie unter User groups (Benutzergruppen) den Namen der **`EmergencyAccessGroup`** aus, die Sie zuvor erstellt haben.
12. Wählen Sie Next (Weiter), um mit der Seite Review and create (Überprüfen und erstellen) fortzufahren.
13. Überprüfen Sie auf der Seite Review and create (Überprüfen und erstellen) die Liste der Benutzergruppenmitgliedschaften, die dem neuen Benutzer hinzugefügt werden sollen. Wenn Sie bereit sind, fortzufahren, wählen Sie Create user (Benutzer erstellen) aus.
14. Wählen Sie auf der Seite Retrieve password (Passwort abrufen) die Option Download .csv file (CSV-Datei herunterladen) aus, um eine CSV-Datei mit den Benutzeranmeldeinformationen (Verbindungs-URL, Benutzername und Passwort) zu speichern.
15. Speichern Sie diese Datei, um sie zu verwenden, wenn Sie sich bei IAM anmelden müssen und keinen Zugriff auf Ihren Verbund-Identitätsanbieter haben.

Der neue IAM-Benutzer wird in der Liste Users (Benutzer) angezeigt. Wählen Sie den Link User name (Benutzername), um die Benutzerdetails anzuzeigen. Kopieren Sie unter Summary (Zusammenfassung) den ARN des Benutzers in die Zwischenablage. Fügen Sie den ARN in ein Textdokument ein, damit Sie ihn im nächsten Verfahren verwenden können.

Erstellen Ihrer ersten Rolle

IAM-Rollen sind eine sichere Methode, um Entitäten, denen Sie vertrauen, Berechtigungen zu gewähren. Es gibt gewisse Ähnlichkeiten zwischen einer IAM-Rolle und einem IAM-Benutzer. Sowohl Rollen als auch Benutzer sind Prinzipale mit Berechtigungsrichtlinien, die bestimmen, was die Identität in AWS tun kann und was nicht. Eine Rolle ist jedoch nicht einer einzigen Person zugeordnet, sondern kann von allen Personen angenommen werden, die diese Rolle benötigen. Einer Rolle sind außerdem keine standardmäßigen, langfristigen Anmeldeinformationen (Passwörter oder Zugriffsschlüssel) zugeordnet. Wenn Sie eine Rolle übernehmen, erhalten Sie stattdessen temporäre Anmeldeinformationen für Ihre Rollensitzung. Die Verwendung von Rollen hilft Ihnen dabei, die bewährten Methoden von IAM zu befolgen. Sie können eine Rolle verwenden, um Folgendes zu tun:

- Ermöglichen Sie Mitarbeiteridentitäten und Identity Center-fähigen Anwendungen den Zugriff auf die AWS Management Console Nutzung AWS IAM Identity Center.
- Delegieren Sie die Erlaubnis an einen AWS Dienst, Aktionen in Ihrem Namen durchzuführen.
- Ermöglichen Sie Anwendungscode, der auf einer Amazon-EC2-Instance ausgeführt wird, um auf AWS -Ressourcen zuzugreifen oder diese zu ändern.
- Gewähren Sie einem anderen AWS-Konto Zugriff.

> **Note**
>
> Sie können AWS Identity and Access Management Roles Anywhere verwenden, um Zugriff auf Maschinenidentitäten zu gewähren. Wenn Sie IAM Roles Anywhere verwenden, müssen Sie keine langfristigen Anmeldeinformationen für Workloads verwalten, die außerhalb von ausgeführt werden. AWS Weitere Informationen finden Sie unter Was ist AWS Identity and Access Management Roles Anywhere? im AWS Identity and Access Management Roles Anywhere-Benutzerhandbuch.

IAM Identity Center und andere AWS Dienste erstellen automatisch Rollen für ihre Dienste. Wenn Sie IAM-Benutzer verwenden, empfehlen wir, Rollen zu erstellen, die Ihre Benutzer bei ihrer Anmeldung annehmen können. Dadurch erhalten sie temporäre Berechtigungen während der Sitzung anstelle von langfristigen Berechtigungen.

Der AWS Management Console Assistent, der Sie durch die Schritte zum Erstellen einer Rolle führt, zeigt leicht unterschiedliche Schritte an, je nachdem, ob Sie eine Rolle für einen IAM-Benutzer, AWS Dienst oder für einen Verbundbenutzer erstellen. Der reguläre Zugriff AWS-Konten innerhalb einer Organisation sollte über den Verbundzugriff erfolgen. Wenn Sie IAM-Benutzer für bestimmte Zwecke erstellen, z. B. für den Notfallzugriff oder den programmatischen Zugriff, gewähren Sie diesen IAM-Benutzern nur die Berechtigung, eine Rolle anzunehmen, und ordnen Sie diese IAM-Benutzer rollenspezifischen Gruppen zu.

In diesem Verfahren erstellen Sie eine Rolle, die dem EmergencyAccess IAM-Benutzer SupportUser Zugriff gewährt. Bevor Sie dieses Verfahren starten, kopieren Sie den ARN des IAM-Benutzers in die Zwischenablage.

So erstellen Sie eine Rolle für einen IAM-Benutzer

1. Folgen Sie dem Anmeldeverfahren, das Ihrem Benutzertyp entspricht, wie im Abschnitt So melden Sie sich bei AWS an im AWS -Anmelde-Benutzerhandbuch beschrieben.
2. Wählen Sie auf der Console Home (Konsolen-Startseite) den IAM-Service aus.
3. Klicken Sie im Navigationsbereich der IAM-Konsole auf Roles (Rollen) und wählen Sie dann Create role (Rolle erstellen).
4. Wählen Sie den Rollentyp AWS-Konto.
5. Wählen Sie in Select trusted entity (Vertrauenswürdige Entität auswählen) unter Trusted entity type (Typ der vertrauenswürdigen Entität) die Option Custom trust policy (Benutzerdefinierte Vertrauensrichtlinie) aus.
6. Prüfen Sie im Abschnitt Custom trust policy (Benutzerdefinierte Vertrauensrichtlinie) die grundlegende Vertrauensrichtlinie. Dies ist die, die wir für diese Rolle verwenden werden. Verwenden Sie den Editor Edit statement (Statement bearbeiten), um die Vertrauensrichtlinie zu aktualisieren.

    1. Wählen Sie unter Add actions for STS (Aktionen für STS hinzufügen) die Option Assume role (Rolle annehmen) aus.
    2. Wählen Sie neben Add a principal (Einen Prinzipal hinzufügen) die Option Add (Hinzufügen) aus. Das Fenster Add principal (Prinzipal hinzufügen) wird geöffnet.

       Wählen Sie unter Prinzipal type (Prinzipaltyp) die Option IAM Users (IAM-Benutzer) aus.

       Fügen Sie unter ARN den ARN des IAM-Benutzers ein, den Sie in die Zwischenablage kopiert haben.

Wählen Sie Add principal (Prinzipal hinzufügen) aus.

3. Stellen Sie sicher, dass die `Principal`-Zeile in der Vertrauensrichtlinie jetzt den von Ihnen angegebenen ARN enthält:

   ```
   "Principal": { "AWS": "arn:aws:iam::123456789012:user/username" }
   ```

7. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter.
8. Wählen Sie in Add permissions (Berechtigungen hinzufügen) unter das Kontrollkästchen neben der anzuwendenden Berechtigungsrichtlinie aus. Für dieses Tutorial wählen wir die SupportUserVertrauensrichtlinie aus. Sie können diese Rolle dann verwenden, um Probleme mit dem zu beheben AWS-Konto und Supportanfragen mit zu eröffnen AWS. Wir legen derzeit keine Berechtigungsgrenze fest.
9. Wählen Sie Weiter aus.
10. Vervollständigen Sie unter Name, review, and create (Benennen, überprüfen und erstellen) diese Einstellungen:

    - Geben Sie im Feld Rollenname einen Namen ein, der diese Rolle identifiziert, z. SupportUserRoleB.
    - Erläutern Sie unter Description (Beschreibung) den Verwendungszweck der Rolle.

    Da andere AWS Ressourcen möglicherweise auf die Rolle verweisen, können Sie den Namen der Rolle nicht bearbeiten, nachdem sie erstellt wurde.
11. Wählen Sie Create rule (Regel erstellen) aus.

    Nachdem Sie die Rolle erstellt haben, teilen Sie die Rolleninformationen mit den Personen, die die Rolle benötigen. Sie können die Rolleninformationen teilen, indem Sie Folgendes tun:

    - Link zur Rolle: Sie können den Benutzern einen Link zusenden, mit dem sie zur Seite Switch Role (Rolle wechseln) gelangen, auf der sämtliche Angaben bereits ausgefüllt sind.
    - Konto-ID oder Alias: Geben Sie jedem Benutzer den Rollennamen sowie die Konto-ID-Nummer oder den Konto-Alias. Der Benutzer öffnet dann die Seite Switch Role (Rolle wechseln) und gibt die Details manuell ein.
    - Speichern der Informationen zum Rollenlink zusammen mit den EmergencyAccess Benutzeranmeldedaten.

Details hierzu finden Sie unter Bereitstellen von Informationen an den Benutzer.

Erstellen Ihrer ersten IAM-Richtlinie

Anschließend fügen Sie die Richtlinien an IAM-Identitäten (Benutzer, Benutzergruppen oder Rollen) oder AWS -Ressourcen an. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert.

So erstellen Sie Ihre erste IAM-Richtlinie

1. Folgen Sie dem Anmeldeverfahren, das Ihrem Benutzertyp entspricht, wie im Abschnitt So melden Sie sich bei AWS an im AWS -Anmelde-Benutzerhandbuch beschrieben.
2. Wählen Sie auf der Console Home (Konsolen-Startseite) den IAM-Service aus.
3. Wählen Sie im Navigationsbereich Policies aus.

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
4. Wählen Sie Create Policy (Richtlinie erstellen) aus.
5. Wählen Sie auf der Seite Richtlinie erstellen die Option Aktionen und wählen Sie dann Richtlinie importieren aus.
6. Geben Sie im Fenster Richtlinie importieren im Feld Richtlinien suchen **`power`** ein, um die Liste der Richtlinien zu reduzieren. Wählen Sie die PowerUserAccessRichtlinie aus.
7. Wählen Sie Richtlinie importieren aus. Die Richtlinie wird auf der Registerkarte JSON angezeigt.
8. Wählen Sie Weiter aus.
9. Geben Sie auf der Seite Überprüfen und Erstellen als Richtliniennamen **`PowerUserExamplePolicy`** ein. Geben Sie in Description (Beschreibung) den Text **`Allows full access to all services except those for user management`** ein. Wählen Sie dann Create policy (Richtlinie erstellen) aus, um die neue Richtlinie zu speichern.

Sie können diese Richtlinie an eine Rolle anhängen, um Benutzern, die diese Rolle übernehmen, die mit dieser Richtlinie verbundenen Berechtigungen zu gewähren. Die PowerUserAccessRichtlinie wird häufig verwendet, um Entwicklern Zugriff zu gewähren.

Programmgesteuerter Zugriff

Benutzer benötigen programmatischen Zugriff, wenn sie mit AWS außerhalb von interagieren möchten. AWS Management Console Die Art und Weise, wie programmatischer Zugriff gewährt wird, hängt von der Art des Benutzers ab, der zugreift: AWS

- Wenn Sie Identitäten in IAM Identity Center verwalten, benötigen die AWS APIs ein Profil und für die AWS Command Line Interface APIs ein Profil oder eine Umgebungsvariable.
- Wenn Sie IAM-Benutzer haben, AWS Command Line Interface benötigen die AWS APIs und die Zugriffsschlüssel. Wenn möglich, erstellen Sie temporäre Anmeldeinformationen, die aus einer Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sicherheits-Token bestehen, das angibt, wann die Anmeldeinformationen ablaufen.

Wählen Sie eine der folgenden Optionen aus, um den Benutzern programmgesteuerten Zugriff zu gewähren.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
Mitarbeiteridentität (Benutzer, die in IAM Identity Center verwaltet werden)	Verwenden Sie kurzfristige Anmeldeinformationen, um programmatische Anfragen an die AWS APIs AWS CLI oder zu signieren (direkt oder mithilfe der AWS SDKs).	Folgen Sie den Anweisungen für die Schnittstelle, die Sie verwenden möchten: • Folgen Sie dazu den AWS CLI Anweisungen unter Abrufen von IAM-Rolle nanmeldedaten für den CLI-Zugriff im AWS IAM Identity Center Benutzerhandbuch. • Folgen Sie für die AWS APIs den Anweisungen unter SSO-Anmeldeinforma tionen im Referenzh andbuch für AWS SDKs und Tools.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
IAM	Verwenden Sie kurzfristige Anmeldeinformationen, um programmatische Anfragen an die AWS APIs AWS CLI oder zu signieren (direkt oder mithilfe der AWS SDKs).	Folgen Sie den Anweisungen unter Temporäre Anmeldeinformationen mit AWS Ressourcen verwenden.
IAM	Verwenden Sie langfristige Anmeldeinformationen, um programmatische Anfragen an die AWS APIs AWS CLI oder zu signieren (direkt oder mithilfe der AWS SDKs). (Nicht empfohlen)	Folgen Sie den Anweisungen unter Verwalten der Zugriffsschlüssel für IAM-Benutzer.

Bewährte Methoden und Anwendungsfälle für Sicherheit in AWS Identity and Access Management

AWS Identity and Access Management (IAM) bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden stellen allgemeine Richtlinien und keine vollständige Sicherheitslösung dar. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

Wenn Sie den größten Nutzen aus IAM ziehen möchten, sollten Sie die empfohlenen bewährten Methoden anwenden. Eine Möglichkeit dazu ist, sich anzusehen, wie IAM in realen Szenarien für die Zusammenarbeit mit AWS -Services verwendet wird.

Themen

Bewährte Methoden für die Sicherheit in IAM

Follow us on Twitter

Die AWS Identity and Access Management Best Practices wurden am 14. Juli 2022 aktualisiert.

Folgen Sie diesen Best Practices für AWS Identity and Access Management (IAM), um Ihre AWS Ressourcen zu schützen.

Themen

- Aktualisieren Sie Zugriffsschlüssel für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern
- Befolgen Sie bewährte Methoden zum Schutz Ihrer Root-Benutzer-Anmeldedaten
- Gewähren Sie die geringsten Berechtigungen
- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten
- Verwenden Sie IAM Access Analyzer, um Richtlinien für die geringste Berechtigung basierend auf der Zugriffsaktivität zu generieren
- Überprüfen und entfernen Sie regelmäßig nicht verwendete Benutzer, Rollen, Berechtigungen, Richtlinien und Anmeldeinformationen
- Verwenden Sie Bedingungen in IAM-Richtlinien, um den Zugriff weiter einzuschränken
- Überprüfen Sie den öffentlichen und kontoübergreifenden Zugriff auf Ressourcen mit IAM Access Analyzer
- Verwenden Sie IAM Access Analyzer, um Ihre IAM-Richtlinien zu validieren und sichere und funktionale Berechtigungen zu gewährleisten
- Richten Sie den Berechtigungs-Integritätsschutz für mehrere Konten ein
- Verwenden Sie Berechtigungsgrenzen, um die Berechtigungsverwaltung innerhalb eines Kontos zu delegieren

Erfordern Sie menschliche Benutzer, einen Verbund mit einem Identitätsanbieter zu verwenden, um mit temporären AWS Anmeldeinformationen darauf zuzugreifen

Menschliche Benutzer, auch bekannt als menschliche Identitäten, sind die Personen, Administratoren, Entwickler, Betreiber und Verbraucher Ihrer Anwendungen. Sie müssen über eine Identität verfügen, um auf Ihre AWS Umgebungen und Anwendungen zugreifen zu können. Menschliche Benutzer, die Mitglieder Ihrer Organisation sind, werden auch als Mitarbeiteridentitäten bezeichnet. Menschliche Benutzer können auch externe Benutzer sein, mit denen Sie zusammenarbeiten und die mit Ihren AWS -Ressourcen interagieren. Sie können dies über einen Webbrowser, eine Client-Anwendung, eine mobile App oder interaktive Befehlszeilentools tun.

Erfordern Sie von Ihren menschlichen Benutzern, beim Zugriff temporäre Anmeldeinformationen zu verwenden AWS. Sie können einen Identitätsanbieter für Ihre menschlichen Benutzer verwenden,

um Verbundzugriff zu gewähren, AWS-Konten indem Sie Rollen übernehmen, die temporäre Anmeldeinformationen bereitstellen. Für eine zentrale Zugriffsverwaltung empfehlen wir Ihnen die Verwendung von AWS IAM Identity Center (IAM Identity Center), um den Zugriff auf Ihre Konten und die Berechtigungen innerhalb dieser Konten zu verwalten. Sie können Ihre Benutzeridentitäten mit IAM Identity Center verwalten oder Zugriffsberechtigungen für Benutzeridentitäten in IAM Identity Center von einem externen Identitätsanbieter verwalten. Weitere Informationen finden Sie unter Was ist AWS IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

Weitere Informationen zu Rollen finden Sie unter Rollenbegriffe und -konzepte.

Erfordern Sie, dass Workloads für den Zugriff temporäre Anmeldeinformationen mit IAM-Rollen verwenden AWS

Ein Workload ist eine Sammlung von Ressourcen und Code, die einen geschäftlichen Nutzen erbringen, z. B. eine Anwendung oder ein Backend-Prozess. Ihre Workload kann Anwendungen, Betriebstools und Komponenten enthalten, die eine Identität erfordern, um Anforderungen an AWS-Serviceszu stellen, z. B. Anforderungen zum Lesen von Daten. Zu diesen Identitäten gehören Maschinen, die in Ihren AWS Umgebungen ausgeführt werden, z. B. Amazon EC2 EC2-Instances oder AWS Lambda -Funktionen.

Sie können auch Maschinenidentitäten für externe Parteien verwalten, die Zugriff benötigen. Um Zugriff auf Maschinenidentitäten zu gewähren, können Sie IAM-Rollen verwenden. IAM-Rollen verfügen über spezifische Berechtigungen und ermöglichen den Zugriff, AWS indem sie sich bei einer Rollensitzung auf temporäre Sicherheitsanmeldedaten verlassen. Darüber hinaus benötigen möglicherweise Computer außerhalb AWS dieser Systeme Zugriff auf Ihre AWS Umgebungen. Für Maschinen, die außerhalb von AWS Ihnen laufen, können Sie AWS Identity and Access Management Roles Anywhere verwenden. Weitere Informationen zu Rollen finden Sie unter IAM-Rollen. Einzelheiten dazu, wie Sie Rollen verwenden können, um den Zugriff an andere zu delegieren AWS-Konten, finden Sie unterTutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen.

Verlangt eine Multi-Faktor-Authentifizierung (MFA)

Wir empfehlen die Verwendung von IAM-Rollen für menschliche Benutzer und Workloads, die auf Ihre AWS -Ressourcen zugreifen, damit diese temporäre Anmeldeinformationen verwenden. Für Szenarien, in denen Sie IAM-Benutzer oder Root-Benutzer in Ihrem Konto benötigen, benötigen Sie jedoch MFA für zusätzliche Sicherheit. Mit MFA verfügen Benutzer über ein System, das eine Antwort auf eine Authentifizierungsaufgabe generiert. Die Anmeldeinformationen und die vom Gerät

generierte Antwort jedes Benutzers sind erforderlich, um den Anmeldevorgang abzuschließen. Weitere Informationen finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

Wenn Sie IAM Identity Center für die zentrale Zugriffsverwaltung für menschliche Benutzer verwenden, können Sie die MFA-Funktionen von IAM Identity Center verwenden, wenn Ihre Identitätsquelle mit dem IAM Identity Center-Identitätsspeicher, AWS Managed Microsoft AD oder AD Connector konfiguriert ist. Weitere Informationen über MFA in IAM Identity Center finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) im AWS IAM Identity Center - Benutzerhandbuch.

Aktualisieren Sie Zugriffsschlüssel für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern

Wenn möglich, empfehlen wir, sich auf temporäre Anmeldeinformationen zu verlassen, anstatt langfristige Anmeldeinformationen wie Zugriffsschlüssel zu erstellen. Für Szenarien, in denen Sie IAM-Benutzer mit programmgesteuertem Zugriff und langfristigen Anmeldeinformationen benötigen, empfehlen wir Ihnen jedoch, die Zugriffsschlüssel bei Bedarf zu aktualisieren, beispielsweise wenn ein Mitarbeiter Ihr Unternehmen verlässt. Wir empfehlen, dass Sie IAM-Zugriff auf zuletzt verwendete Informationen verwenden, um Zugriffsschlüssel sicher zu aktualisieren und zu entfernen. Weitere Informationen finden Sie unter Aktualisierung der Zugriffsschlüssel.

Es gibt spezielle Anwendungsfälle, die langfristige Anmeldeinformationen mit IAM-Benutzern in AWS erfordern. Einige der Anwendungsfälle umfassen Folgendes:

- Workloads, die IAM-Rollen nicht verwenden können – Sie könnten einen Workload von einem Speicherort ausführen, der auf AWS zugreifen muss. In einigen Situationen können Sie IAM-Rollen nicht verwenden, um temporäre Anmeldeinformationen bereitzustellen, z. B. für Plugins. WordPress Verwenden Sie in diesen Situationen langfristige IAM-Benutzerzugriffsschlüssel für diesen Workload, um sich bei AWS zu authentifizieren.
- AWS Drittanbieter-Clients — Wenn Sie Tools verwenden, die den Zugriff mit IAM Identity Center nicht unterstützen, z. B. AWS Drittanbieter-Clients oder Anbieter, die nicht auf gehostet werden AWS, verwenden Sie langfristige IAM-Benutzerzugriffsschlüssel.
- AWS CodeCommit Zugriff — Wenn Sie Ihren Code CodeCommit zum Speichern verwenden, können Sie einen IAM-Benutzer mit SSH-Schlüsseln oder dienstspezifischen Anmeldeinformationen verwenden, um sich bei Ihren Repositorys CodeCommit zu authentifizieren. Wir empfehlen Ihnen, dies zusätzlich zu einem Benutzer im IAM Identity Center für die normale Authentifizierung zu verwenden. Benutzer in IAM Identity Center sind die Personen in Ihrer

Belegschaft, die Zugriff auf Ihre oder Ihre AWS-Konten Cloud-Anwendungen benötigen. Um Benutzern Zugriff auf Ihre CodeCommit Repositorys zu gewähren, ohne IAM-Benutzer zu konfigurieren, können Sie das Hilfsprogramm konfigurieren. git-remote-codecommit Weitere Informationen zu IAM und CodeCommit finden Sie unter. Verwenden von IAM mit CodeCommit: Git-Anmeldeinformationen, SSH-Schlüsseln und AWS Zugriffsschlüsseln Weitere Informationen zur Konfiguration des git-remote-codecommit Dienstprogramms finden Sie im AWS CodeCommit Benutzerhandbuch unter Herstellen einer Verbindung zu AWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen.

- Zugriff auf Amazon Keyspaces (für Apache Cassandra) – In einer Situation, in der Sie Benutzer im IAM Identity Center nicht verwenden können, z. B. zu Testzwecken für die Cassandra-Kompatibilität, können Sie einen IAM-Benutzer mit dienstspezifischen Anmeldeinformationen verwenden, um sich bei Amazon Keyspaces zu authentifizieren. Benutzer in IAM Identity Center sind die Personen in Ihrer Belegschaft, die Zugriff auf Ihre AWS-Konten oder Ihre Cloud-Anwendungen benötigen. Sie können auch mithilfe temporärer Anmeldeinformationen eine Verbindung zu Amazon Keyspaces herstellen. Weitere Informationen finden Sie unter Verwendung temporärer Anmeldeinformationen für die Verbindung zu Amazon Keyspaces mithilfe einer IAM-Rolle und des SigV4-Plugins im Amazon Keyspaces (für Apache Cassandra)-Entwicklerhandbuch.

Befolgen Sie bewährte Methoden zum Schutz Ihrer Root-Benutzer-Anmeldedaten

Wenn Sie eine erstellen AWS-Konto, richten Sie Root-Benutzeranmeldedaten ein, um sich bei der AWS Management Console anzumelden. Schützen Sie Ihre Root-Benutzeranmeldeinformationen auf die gleiche Weise wie andere vertrauliche persönliche Daten. Weitere Informationen zur Sicherung und Skalierung Ihrer Root-Benutzer-Prozesse finden Sie unter Bewährte Methoden für Root-Benutzer für Ihren AWS-Konto.

Gewähren Sie die geringsten Berechtigungen

Gewähren Sie die Berechtigungen mit IAM Richtlinien nur die zum Ausführen 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. Sie können mit umfassenden Berechtigungen beginnen, während Sie die für Ihren Workload oder Anwendungsfall erforderlichen Berechtigungen untersuchen. Mit zunehmender Reife Ihres Anwendungsfalls können Sie daran arbeiten, die Berechtigungen zu reduzieren, die Sie gewähren, um auf die geringsten Berechtigungen hinzuarbeiten. Weitere Informationen zur

Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Berechtigungen und Richtlinien in IAM.

Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten

Um Ihren Benutzern und Workloads Berechtigungen zu gewähren, verwenden Sie die AWS -verwaltete Richtlinien die Berechtigungen für viele häufige Anwendungsfälle gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie für alle Kunden verfügbar sind. AWS Daher empfehlen wir Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete Richtlinien definieren, die spezifisch auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS verwaltete Richtlinien. Weitere Informationen zu AWS verwalteten Richtlinien, die für bestimmte Aufgabenbereiche konzipiert sind, finden Sie unter. AWS verwaltete Richtlinien für Jobfunktionen

Verwenden Sie IAM Access Analyzer, um Richtlinien für die geringste Berechtigung basierend auf der Zugriffsaktivität zu generieren

Um nur die zum Ausführen einer Aufgabe erforderlichen Berechtigungen zu erteilen, können Sie Richtlinien auf der Grundlage Ihrer in AWS CloudTrail protokollierten Zugriffsaktivitäten erstellen. IAM Access Analyzer analysiert die Dienste und Aktionen, die Ihre IAM-Rollen verwenden, und generiert dann eine fein abgestimmte Richtlinie, die Sie verwenden können. Nachdem Sie jede generierte Richtlinie getestet haben, können Sie die Richtlinie in Ihrer Produktionsumgebung bereitstellen. Dadurch wird sichergestellt, dass Sie nur die erforderlichen Berechtigungen für Ihre Workloads gewähren. Weitere Informationen zur Richtlinienerstellung finden Sie unter IAM Access Analyzer Richtlinienerstellung.

Überprüfen und entfernen Sie regelmäßig nicht verwendete Benutzer, Rollen, Berechtigungen, Richtlinien und Anmeldeinformationen

Möglicherweise haben Sie IAM-Benutzer, -Rollen, -Berechtigungen, -Richtlinien oder -Berechtigungsnachweise, die Sie in Ihrem AWS-Konto nicht mehr benötigen. IAM stellt Informationen zum letzten Zugriff zur Verfügung, mit deren Hilfe Sie Benutzer, Rollen, Berechtigungen, Richtlinien und Anmeldeinformationen identifizieren können, die Sie nicht mehr benötigen, damit Sie sie entfernen können. Auf diese Weise können Sie die Anzahl der Benutzer, Rollen, Berechtigungen, Richtlinien und Anmeldeinformationen, die Sie überwachen müssen, reduzieren. Sie können

diese Informationen auch dazu verwenden, Ihre IAM-Richtlinien so zu verfeinern, sodass sie die Berechtigungen mit den geringsten Berechtigungen besser einhalten. Weitere Informationen finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Verwenden Sie Bedingungen in IAM-Richtlinien, um den Zugriff weiter einzuschränken

Sie können die Bedingungen angeben, unter denen eine Richtlinienanweisung wirksam ist. Auf diese Weise können Sie Zugriff auf Aktionen und Ressourcen gewähren, jedoch nur, wenn die Zugriffsanfrage bestimmte Bedingungen erfüllt. 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, aber nur, wenn sie über einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Condition.

Überprüfen Sie den öffentlichen und kontoübergreifenden Zugriff auf Ressourcen mit IAM Access Analyzer

Bevor Sie Berechtigungen für den öffentlichen oder kontoübergreifenden Zugriff in gewähren AWS, empfehlen wir Ihnen, zu überprüfen, ob ein solcher Zugriff erforderlich ist. Sie können IAM Access Analyzer verwenden, um eine Vorschau und Analyse des öffentlichen und kontoübergreifenden Zugriffs für unterstützte Ressourcentypen zu erhalten. Sie tun dies, indem Sie die Befunde überprüfen, die IAM Access Analyzer generiert. Mithilfe dieser Ergebnisse können Sie überprüfen, ob Ihre Ressourcenzugriffskontrollen den erwarteten Zugriff gewähren. Wenn Sie öffentliche und kontoübergreifende Berechtigungen aktualisieren, können Sie außerdem die Auswirkungen Ihrer Änderungen überprüfen, bevor Sie neue Zugriffskontrollen für Ihre Ressourcen bereitstellen. IAM Access Analyzer überwacht außerdem kontinuierlich unterstützte Ressourcentypen und generiert eine Suche nach Ressourcen, die einen öffentlichen oder kontoübergreifenden Zugriff ermöglichen. Weitere Informationen finden Sie unter Vorschau des Zugriffs mit IAM Access Analyzer APIs.

Verwenden Sie IAM Access Analyzer, um Ihre IAM-Richtlinien zu validieren und sichere und funktionale Berechtigungen zu gewährleisten

Validieren Sie die Richtlinien, die Sie erstellen, um sicherzustellen, dass sie der IAM-Richtliniensprache (JSON) und den bewährten Methoden von IAM entsprechen. Sie können Ihre Richtlinien mithilfe der Richtlinienvalidierung von IAM Access Analyzer validieren. IAM Access

Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Während Sie neue Richtlinien erstellen oder bestehende Richtlinien in der Konsole bearbeiten, bietet IAM Access Analyzer Empfehlungen, die Ihnen helfen, Ihre Richtlinien zu verfeinern und zu validieren, bevor Sie sie speichern. Zusätzlich empfehlen wir Ihnen, alle Ihre bestehenden Richtlinien zu überprüfen und zu validieren. Weitere Informationen finden Sie unter IAM Access Analyzer Richtlinienvalidierung. Weitere Informationen zu den von IAM Access Analyzer bereitgestellten Richtlinienprüfungen finden Sie unter Referenz für Richtlinienprüfungen zu IAM Access Analyzer.

Richten Sie den Berechtigungs-Integritätsschutz für mehrere Konten ein

Wenn Sie Ihre Workloads skalieren, trennen Sie sie voneinander, indem Sie mehrere Konten verwenden, die mit verwaltet werden. AWS Organizations Wir empfehlen, dass Sie die Service-Kontrollrichtlinien (SCPs) der Organisation verwenden, um den Berechtigungs-Integritätsschutz einzurichten, um den Zugriff für alle IAM-Benutzer und -Rollen in Ihren Konten zu kontrollieren. SCPs sind eine Art von Organisationsrichtlinie, mit der Sie Berechtigungen in Ihrer Organisation auf Organisations-, OU- oder Kontoebene verwalten können. AWS Der von Ihnen festgelegten Berechtigungs-Integritätsschutz, den Sie einrichten, gilt für alle Benutzer und Rollen innerhalb der abgedeckten Konten. SCPs allein reichen jedoch nicht aus, um den Konten in Ihrer Organisation Berechtigungen zu gewähren. Dazu muss Ihr Administrator identitätsbasierte oder ressourcenbasierte Richtlinien an IAM-Benutzer, IAM-Rollen oder die Ressourcen in Ihren Konten anfügen. Weitere Informationen finden Sie unter AWS Organizations, Konten und IAM-Integritätsschutz.

Verwenden Sie Berechtigungsgrenzen, um die Berechtigungsverwaltung innerhalb eines Kontos zu delegieren

In einigen Szenarios kann es sinnvoll sein, die Berechtigungsverwaltung innerhalb eines Kontos an andere zu delegieren. Sie könnten Entwicklern beispielsweise ermöglichen, Rollen für ihre Workloads zu erstellen und zu verwalten. Wenn Sie Berechtigungen an andere delegieren, verwenden Sie Berechtigungsgrenzen, um die maximalen Berechtigungen festzulegen, die Sie delegieren. Eine Berechtigungsgrenze ist ein erweitertes Feature, bei der Sie eine verwaltete Richtlinie verwenden, um die maximalen Berechtigungen festzulegen, die eine identitätsbasierte Richtlinie einer IAM-Rolle gewähren kann. Eine Berechtigungsgrenze gewährt selbst keine Berechtigungen. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten.

Bewährte Methoden für Root-Benutzer für Ihren AWS-Konto

Wenn Sie ein Konto zum ersten Mal erstellen AWS-Konto, verwenden Sie zunächst einen Satz von Standardanmeldedaten mit vollständigem Zugriff auf alle AWS Ressourcen in Ihrem Konto. Diese Identität wird als AWS-Konto -Root-Benutzer bezeichnet. Wir empfehlen dringend, dass Sie nicht auf den AWS-Konto Root-Benutzer zugreifen, es sei denn, Sie haben eine Aufgabe, für die Root-Benutzeranmeldedaten erforderlich sind. Sie müssen Ihre Root-Benutzeranmeldeinformationen und die Mechanismen zur Kontowiederherstellung sichern, um sicherzustellen, dass Sie Ihre Anmeldeinformationen mit weitreichenden Berechtigungen nicht für unbefugte Zwecke preisgeben.

Anstatt auf den Root-Benutzer zuzugreifen, erstellen Sie einen Administratorbenutzer für alltägliche Aufgaben.

- Eine einzelne, eigenständige Version AWS-Konto finden Sie unterErstellen Sie einen Benutzer mit Administratorzugriff.
- Informationen zur AWS-Konten Verwaltung mehrerer Dateien AWS Organizations finden Sie unter AWS-Konto Zugriff für einen IAM Identity Center-Administratorbenutzer einrichten.

Mit Ihrem administrativen Benutzer können Sie dann zusätzliche Identitäten für Benutzer erstellen, die Zugriff auf Ressourcen in Ihrem AWS-Konto benötigen. Wir empfehlen dringend, dass Benutzer sich beim Zugriff mit temporären Anmeldeinformationen authentifizieren müssen. AWS

- Verwenden Sie diese Option für eine einzelne AWS-Konto, eigenständige Version, IAM-Rollen um Identitäten in Ihrem Konto mit bestimmten Berechtigungen zu erstellen. Rollen sollten von jedem übernommen werden können, der sie benötigt. Einer Rolle sind außerdem keine standardmäßigen, langfristigen Anmeldeinformationen (Passwörter oder Zugriffsschlüssel) zugeordnet. Wenn Sie eine Rolle übernehmen, erhalten Sie stattdessen temporäre Anmeldeinformationen für Ihre Rollensitzung. Im Gegensatz zu IAM-Rollen verfügen IAM-Benutzer über langfristige Anmeldeinformationen wie Passwörter und Zugangsschlüssel. Wenn möglich, empfehlen die bewährten Methoden, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben.
- Verwenden Sie für mehrere über Organizations AWS-Konten verwaltete IAM Identity Center Workforce-Benutzer. Mit IAM Identity Center können Sie alle Benutzer AWS-Konten und die Berechtigungen innerhalb dieser Konten zentral verwalten. Verwalten Sie Ihre Benutzeridentitäten mit IAM Identity Center oder von einem externen Identitätsanbieter. Weitere Informationen finden Sie unter Was ist AWS IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

Themen

Schützen Sie Ihre Root-Benutzer-Anmeldedaten, um eine unbefugte Nutzung zu verhindern

Schützen Sie Ihre Root-Benutzeranmeldeinformationen und verwenden Sie sie nur für die Aufgaben, die sie erfordern. Um eine unbefugte Nutzung zu verhindern, geben Sie Ihr Root-Benutzerpasswort, Ihre MFA, Ihre Zugriffsschlüssel, CloudFront Schlüsselpaare oder Ihre Signaturzertifikate nicht an Dritte weiter, außer an Personen, die aus geschäftlichen Gründen auf den Root-Benutzer zugreifen müssen.

Speichern Sie das Root-Benutzerkennwort nicht zusammen mit Tools, die von AWS-Services einem Konto abhängen, auf das mit demselben Passwort zugegriffen wird. Wenn Sie Ihr Root-Benutzerpasswort verlieren oder vergessen, können Sie nicht auf diese Tools zugreifen. Wir empfehlen Ihnen, der Ausfallsicherheit Priorität einzuräumen und zu erwägen, dass zwei oder mehr Personen den Zugriff auf den Speicherort autorisieren müssen. Der Zugriff auf das Passwort oder dessen Speicherort sollten protokolliert und überwacht werden.

Verwenden eines sicheren Root-Benutzerpassworts zum Schutz des Zugriffs

Es empfiehlt sich, ein sicheres und eindeutiges Passwort zu verwenden. Tools wie Passwort-Manager mit starken Algorithmen zur Passwortgenerierung können Ihnen dabei helfen, diese Ziele zu erreichen. AWS setzt voraus, dass Ihr Passwort die folgenden Bedingungen erfüllt:

- Es muss mindestens 8 Zeichen und maximal 128 Zeichen lang sein.
- Es muss mindestens drei der folgenden Zeichentypen enthalten: Großbuchstaben, Kleinbuchstaben, Zahlen und ! @ # $ % ^ & * () <> [] {} | _+-= Symbole.
- Es darf nicht mit Ihrem AWS-Konto Namen oder Ihrer E-Mail-Adresse identisch sein.

Weitere Informationen finden Sie unter Ändern Sie das Passwort für Root-Benutzer des AWS-Kontos.

Sichern Ihren Stammbenutzeranmeldung mit Multi-Faktor-Authentifizierung (MFA)

Da ein Root-Benutzer privilegierte Aktionen ausführen kann, ist es wichtig, zusätzlich zur E-Mail-Adresse und dem Passwort als Anmeldedaten, dem Root-Benutzer MFA als zweiten Authentifizierungsfaktor hinzuzufügen. Wir empfehlen ausdrücklich, mehrere MFA für Ihre Root-Benutzer-Anmeldedaten zu aktivieren, um Ihrer Sicherheitsstrategie zusätzliche Flexibilität und Stabilität zu verleihen. Sie können bis zu acht MFA-Geräte einer beliebigen Kombination der derzeit unterstützten MFA-Typen für Ihren AWS-Konto -Root-Benutzer registrieren.

- FIDO-zertifizierte Hardware-Sicherheitsschlüssel werden von Drittanbietern bereitgestellt. Weitere Informationen finden Sie unter Aktivieren eines FIDO-Sicherheitsschlüssels für den AWS-Konto Root-Benutzer.
- Ein Hardwaregerät, das auf der Grundlage des Algorithmus für ein zeitgesteuertes Einmalpasswort (TOTP) einen sechsstelligen numerischen Code generiert. Weitere Informationen finden Sie unter Aktivieren eines Hardware-TOTP-Tokens für den AWS-Konto Root-Benutzer.
- Eine virtuelle Authentifizierungsanwendung, die auf einem Telefon oder einem anderen Gerät ausgeführt wird und ein physisches Gerät emuliert. Weitere Informationen finden Sie unter Aktivieren eines virtuellen MFA-Geräts für Ihren AWS-Konto Root-Benutzer.

Keine Zugriffsschlüssel für den Root-Benutzer erstellen

Mit den Zugriffstasten können Sie Befehle in der AWS Befehlszeilenschnittstelle (AWS CLI) ausführen oder API-Operationen von einem der AWS SDKs aus verwenden. Es wird dringend empfohlen, keine Zugriffsschlüsselpaare für Ihren Root-Benutzer zu erstellen, da der Root-Benutzer vollen Zugriff auf alle AWS-Services Ressourcen im Konto hat, einschließlich der Rechnungsinformationen.

Da nur für wenige Aufgaben der Root-Benutzer erforderlich ist und Sie diese Aufgaben normalerweise selten ausführen, empfehlen wir, sich bei dem anzumelden, um Root-Benutzeraufgaben ausführen AWS Management Console zu können. Bevor Sie Zugriffsschlüssel erstellen, prüfen Sie die Alternativen zu Langzeit-Zugriffsschlüsseln.

Verwenden Sie nach Möglichkeit die Genehmigung mehrerer Personen für die Anmeldung als Root-Benutzer

Erwägen Sie, die Genehmigung durch mehrere Personen zu verwenden, um sicherzustellen, dass keine Person sowohl auf MFA als auch auf das Passwort für den Root-Benutzer zugreifen kann. Einige Unternehmen fügen eine zusätzliche Sicherheitsebene hinzu, indem sie eine Gruppe von Administratoren mit Zugriff auf das Passwort und eine weitere Gruppe von Administratoren mit Zugriff auf MFA einrichten. Ein Mitglied aus jeder Gruppe muss sich als Root-Benutzer anmelden.

Verwenden einer Gruppen-E-Mail-Adresse für Root-Benutzer-Anmeldeinformationen

Verwenden Sie eine E-Mail-Adresse, die von Ihrem Unternehmen verwaltet wird und empfangene Nachrichten direkt an eine Gruppe von Benutzern weiterleitet. Wenn Sie den Kontoinhaber kontaktieren AWS müssen, reduziert dieser Ansatz das Risiko von Verzögerungen bei der Beantwortung, selbst wenn Personen im Urlaub sind, krank sind oder das Unternehmen verlassen haben. Die E-Mail-Adresse, die für den Root-Benutzer verwendet wird, sollte nicht für andere Zwecke verwendet werden.

Einschränken des Zugriffs auf Mechanismen zur Kontowiederherstellung

Stellen Sie sicher, dass Sie einen Prozess zur Verwaltung der Wiederherstellungsmechanismen für Root-Benutzer-Anmeldeinformationen entwickeln, für den Fall, dass Sie in Notfällen wie der Übernahme Ihres Administratorkontos darauf zugreifen müssen.

- Stellen Sie sicher, dass Sie Zugriff auf Ihr E-Mail-Postfach für Root-Benutzer haben, damit Sie ein verlorenes oder vergessenes Root-Benutzer-Passwort zurücksetzen können.
- Wenn MFA für Ihren AWS-Konto Root-Benutzer verloren geht, beschädigt ist oder nicht funktioniert, können Sie sich mit einem anderen MFA anmelden, das mit denselben Root-Benutzeranmeldeinformationen registriert ist. Wenn Sie den Zugriff auf all Ihre MFAs verloren haben, benötigen Sie sowohl die Telefonnummer als auch die E-Mail-Adresse, mit der Sie Ihr Konto registriert haben, um auf dem neuesten Stand zu sein und Ihr MFA wiederherzustellen. Einzelheiten finden Sie unter Wiederherstellen eines Root-Benutzer-MFA-Geräts..

- Wenn Sie Ihr Root-Benutzer-Passwort und Ihre MFA nicht speichern möchten, kann die in Ihrem Konto registrierte Telefonnummer als alternative Methode zur Wiederherstellung der Root-Benutzer-Anmeldeinformationen verwendet werden. Stellen Sie sicher, dass Sie Zugriff auf die Kontakttelefonnummer haben, halten Sie die Telefonnummer auf dem neuesten Stand und schränken Sie ein, wer Zugriff auf die Verwaltung der Telefonnummer hat.

Niemand sollte Zugriff sowohl auf den E-Mail-Posteingang sowie auf die Telefonnummer haben, da es sich bei beiden um Bestätigungskanäle handelt, mit denen Sie Ihr Root-Benutzer-Passwort wiederherstellen können. Es ist wichtig, dass zwei Personengruppen diese Kanäle verwalten. Eine Gruppe sollte Zugriff auf Ihre primäre E-Mail-Adresse haben und eine andere Gruppe sollte Zugriff auf die primäre Telefonnummer haben, um den Zugriff auf Ihr Konto als Root-Benutzer wiederherzustellen.

Sichern von Root-Benutzer-Anmeldedaten für Ihr Unternehmenskonto

Wenn Sie bei Organizations zu einer Strategie mit mehreren Konten übergehen, AWS-Konten hat jedes Ihrer eigenen Root-Benutzeranmeldedaten, die Sie sichern müssen. Das Konto, mit dem Sie Ihre Organisation erstellen, ist das Verwaltungskonto, und die übrigen Konten in Ihrer Organisation sind Mitgliedskonten.

Sichern von Root-Benutzer-Anmeldeinformationen für Mitgliedskonten

Wenn Sie Organisationnen zur Verwaltung mehrerer Konten verwenden, gibt es zwei Strategien, die Sie anwenden können, um den Zugriff von Root-Benutzern in Ihren Organisationen zu sichern.

- Sichern Sie die Root-Benutzer-Anmeldeinformationen Ihrer Organisationskonten mit MFA.
- Setzen Sie das Root-Benutzerkennwort für Ihre Konten nicht zurück und stellen Sie den Zugriff darauf nur dann wieder her, wenn dies mithilfe des Verfahrens zur Passwort-Rücksetzung erforderlich ist. Wenn Sie ein Mitgliedskonto in Ihrer Organisation erstellen, erstellt Organisationen automatisch eine IAM-Rolle in dem Mitgliedskonto, die dem Verwaltungskonto den temporären Zugriff auf das Mitgliedskonto ermöglicht.

Einzelheiten finden Sie unter Zugreifen auf Mitgliedskonten in Ihrer Organisation im Organisationen-Benutzerhandbuch.

Einrichten von präventiven Sicherheitskontrollen in Organisationen mithilfe einer Service-Kontrollrichtlinie (SCP)

Wenn Sie Organisationen zum Verwalten mehrerer Konten verwenden, können Sie einen SCP anwenden, um den Zugriff auf den Root-Benutzer des Mitgliedskontos einzuschränken. Das Verweigern aller Root-Benutzeraktionen in Ihren Mitgliedskonten, mit Ausnahme bestimmter Root-Aktionen, trägt dazu bei, unbefugten Zugriff zu verhindern. Einzelheiten finden Sie unter Verwenden eines SCP zum Einschränken der Aktionen des Root-Benutzers in Ihren Mitgliedskonten.

Überwachung von Zugang und Nutzung

Wir empfehlen Ihnen, Ihre derzeitigen Nachverfolgungsmechanismen zu verwenden, um die Anmeldung und Nutzung von Anmeldeinformationen von Root-Benutzern zu überwachen, zu warnen und zu melden, einschließlich Warnungen, die die Anmeldung und Nutzung von Root-Benutzern ankündigen. Die folgenden Services können dazu beitragen, sicherzustellen, dass die Nutzung der Anmeldeinformationen des Root-Benutzers nachverfolgt wird, und Sicherheitsprüfungen durchzuführen, die dazu beitragen können, eine unbefugte Nutzung zu verhindern.

- Wenn Sie über die Anmeldeaktivitäten von Root-Benutzern in Ihrem Konto informiert werden möchten, können Sie Amazon nutzen, um eine Ereignisregel CloudWatch zu erstellen, die erkennt, wenn Root-Benutzeranmeldedaten verwendet werden, und eine Benachrichtigung an Ihren Sicherheitsadministrator auslöst. Einzelheiten finden Sie unter AWS-Konto Root-Benutzeraktivitäten überwachen und benachrichtigen.
- Wenn Sie Benachrichtigungen einrichten möchten, um Sie über genehmigte Root-Benutzeraktionen zu informieren, können Sie Amazon EventBridge zusammen mit Amazon SNS nutzen, um eine EventBridge Regel zu schreiben, um die Nutzung von Root-Benutzern für die jeweilige Aktion zu verfolgen und Sie über ein Amazon SNS SNS-Thema zu benachrichtigen. Ein Beispiel finden Sie unter Senden einer Benachrichtigung, wenn ein Amazon-S3-Objekt erstellt wird.
- Wenn Sie den Dienst zur Bedrohungserkennung GuardDuty bereits verwenden, können Sie dessen Funktion erweitern, sodass Sie benachrichtigt werden, wenn Root-Benutzeranmeldedaten in Ihrem Konto verwendet werden.

Die Warnmeldungen sollten unter anderem die E-Mail-Adresse des Root-Benutzers enthalten. Halten Sie Verfahren für die Reaktion auf Warnungen bereit, damit Mitarbeiter, die eine Warnung zum Root-Benutzerzugriff erhalten, verstehen, wie sie überprüfen können, ob Root-Benutzerzugriff erwartet wird, und wie sie eskalieren können, wenn sie glauben, dass ein Sicherheitsvorfall vorliegt. Ein

Beispiel für die Konfiguration von Warnmeldungen finden Sie unter Überwachen und Benachrichtigen der Root-Benutzeraktivität des AWS-Konto.

Auswertung der MFA-Compliance von Root-Benutzern

- AWS Config verwendet Regeln, um bewährte Methoden für Root-Benutzer durchzusetzen. Sie können AWS verwaltete Regeln verwenden, um zu verlangen, dass Root-Benutzer die Multi-Faktor-Authentifizierung (MFA) aktiviert haben. AWS Config kann auch Zugriffsschlüssel für den Root-Benutzer identifizieren.
- Security Hub bietet Ihnen einen umfassenden Überblick über Ihren Sicherheitsstatus AWS und hilft Ihnen dabei, Ihre AWS Umgebung anhand von Industriestandards und Best Practices zu bewerten, z. B. wenn Sie MFA für den Root-Benutzer haben und keine Root-Benutzerzugriffsschlüssel haben. Einzelheiten zu den verfügbaren Regeln finden Sie unter AWS Identity and Access Management - Steuerelemente im Security-Hub-Benutzerhandbuch.
- Trusted Advisor bietet eine Sicherheitsüberprüfung, damit Sie wissen, ob MFA für das Root-Benutzerkonto nicht aktiviert ist. Weitere Informationen finden Sie unter MFA im Root-Konto im AWS -Support-Benutzerhandbuch.

Wenn Sie ein Sicherheitsproblem in Ihrem Konto melden möchten, lesen Sie Verdächtige E-Mails melden oder Meldung von Schwachstellen. Alternativ können Sie sich an AWS wenden, um Unterstützung und zusätzliche Hilfestellung zu erhalten.

Unternehmensanwendungsfälle für IAM

Ein einfacher geschäftlicher Anwendungsfall für IAM kann Ihnen helfen, grundlegende Möglichkeiten zu verstehen, wie Sie den Service implementieren können, um den AWS Zugriff Ihrer Benutzer zu kontrollieren. Der Anwendungsfall wird allgemein beschrieben, ohne auf die Mechanik einzugehen, wie die IAM-API zum Erreichen der gewünschten Ergebnisse eingesetzt wird.

Dieser Anwendungsfall prüft zwei typische Möglichkeiten, wie ein fiktives Unternehmen mit dem Namen Example Corp IAM verwenden kann. Das erste Szenario betrachtet Amazon Elastic Compute Cloud (Amazon EC2). Die zweite betrachtet Amazon Simple Storage Service (Amazon S3).

Weitere Informationen zur Verwendung von IAM mit anderen Diensten von finden Sie AWS unter. AWS Dienste, die mit IAM funktionieren

Themen

Ersteinrichtung von Example Corp

Nikki Wolf und Mateo Jackson sind die Gründer von Example Corp. Nach der Gründung des Unternehmens erstellen sie ein AWS-Konto und richten es ein AWS IAM Identity Center(IAM Identity Center), um Administratorkonten einzurichten, die sie mit ihren Ressourcen verwenden können. AWS Wenn Sie den Kontozugriff für den Administratorbenutzer einrichten, erstellt IAM Identity Center eine entsprechende IAM-Rolle. Diese Rolle, die vom IAM Identity Center gesteuert wird, wird in der entsprechenden Datei erstellt AWS-Konto, und die im AdministratorAccessBerechtigungssatz angegebenen Richtlinien werden der Rolle zugewiesen.

Da sie jetzt Administratorkonten haben, müssen Nikki und Mateo ihren Root-Benutzer nicht mehr verwenden, um auf ihr AWS-Konto zuzugreifen. Sie planen, den Root-Benutzer nur für Aufgaben zu verwenden, die nur der Root-Benutzer ausführen kann. Nachdem sie die bewährten Sicherheitsmethoden durchgesehen haben, konfigurieren sie die Multi-Faktor-Authentifizierung (MFA) für ihre Root-Benutzeranmeldeinformationen und entscheiden, wie sie ihre Root-Benutzeranmeldeinformationen schützen.

Während ihr Unternehmen wächst, stellen sie Mitarbeiter ein, die als Entwickler, Administratoren, Tester, Manager und Systemadministratoren arbeiten. Nikki ist für den Betrieb verantwortlich, während Mateo die Engineering-Teams leitet. Sie richten einen Active-Directory-Domänenserver ein, um die Mitarbeiterkonten und den Zugriff auf interne Unternehmensressourcen zu verwalten.

Um ihren Mitarbeitern Zugriff auf AWS Ressourcen zu gewähren, verwenden sie IAM Identity Center, um das Active Directory ihres Unternehmens mit ihren zu verbinden. AWS-Konto

Da sie Active Directory mit IAM Identity Center verbunden haben, werden die Benutzer, Gruppen und Gruppenmitgliedschaften synchronisiert und definiert. Sie müssen den verschiedenen Gruppen Berechtigungssätze und Rollen zuweisen, um den Benutzern den richtigen Zugriff auf AWS Ressourcen zu gewähren. Sie verwenden AWS verwaltete Richtlinien für Jobfunktionen in AWS Management Console , um diese Berechtigungssätze zu erstellen:

- Administrator
- Fakturierung

- Entwickler
- Netzwerkadministratoren
- Datenbankadministratoren
- Systemadministratoren
- Support-Benutzer

Anschließend weisen sie diese Berechtigungssätze den Rollen zu, die ihren Active Directory-Gruppen zugewiesen sind.

Eine step-by-step Anleitung zur Erstkonfiguration von IAM Identity Center finden Sie unter Erste Schritte im AWS IAM Identity Center Benutzerhandbuch. Weitere Informationen zur Bereitstellung des IAM Identity Center-Benutzerzugriffs finden Sie im AWS IAM Identity Center -Benutzerhandbuch unter Single Sign-On (SSO)-Zugriff auf AWS -Konten.

Anwendungsfall für IAM mit Amazon EC2

Ein Unternehmen wie Example Corp verwendet IAM in der Regel für die Interaktion mit Diensten wie Amazon EC2. Um diesen Teil des Anwendungsfalls zu verstehen, müssen Sie ein grundlegendes Verständnis von Amazon EC2 haben. Weitere Informationen zu Amazon EC2 finden Sie im Amazon EC2 EC2-Benutzerhandbuch.

Amazon EC2 Berechtigungen für die Benutzergruppen

Um eine „Perimeterkontrolle“ zu gewährleisten, fügt Nikki der Benutzergruppe eine Richtlinie hinzu. AllUsers Diese Richtlinie lehnt jede AWS Anfrage eines Benutzers ab, wenn sich die ursprüngliche IP-Adresse außerhalb des Unternehmensnetzwerks von Example Corp befindet.

Bei Example Corp erfordern verschiedene Gruppen unterschiedliche Berechtigungen:

- System Administrators – Benötigen eine Berechtigung zum Erstellen und Verwalten von AMIs, Instances, Snapshots, Volumes, Sicherheitsgruppen und so weiter. Nikki fügt die `AmazonEC2FullAccess` AWS verwaltete Richtlinie der SysAdmins Benutzergruppe hinzu, die Mitgliedern der Gruppe die Erlaubnis gibt, alle Amazon EC2 EC2-Aktionen zu verwenden.
- Developers – Müssen nur mit Instances arbeiten können. Daher erstellt Mateo eine Richtlinie und fügt sie an die Gruppe „Developers“ an, mit der Entwickler `DescribeInstances`, `RunInstances`, `StopInstances`, `StartInstances` und `TerminateInstances` aufrufen können.

> **Note**
>
> Amazon EC2 verwendet SSH-Schlüssel, Windows-Passwörter und Sicherheitsgruppen, um zu kontrollieren, wer Zugriff auf das Betriebssystem bestimmter Amazon EC2-Instances hat. Es gibt keine Methode im IAM-System, um Zugriff auf das Betriebssystem einer bestimmten Instance zu erlauben oder zu verweigern.

- Support-Benutzer – Sollten nicht in der Lage sein, Amazon EC2-Instances durchzuführen, mit Ausnahme des Auflistens der aktuell verfügbaren Amazon EC2-Ressourcen. Daher erstellt Nikki eine Richtlinie für die Benutzergruppe „Support-Benutzer“, die nur den Aufruf der Amazon-EC2-API-Vorgänge „Beschreiben“ erlaubt.

Beispiele dafür, wie diese Richtlinien aussehen könnten, finden Sie unter Beispiele für identitätsbasierte Richtlinien in IAM und Using AWS Identity and Access Management im Amazon EC2 EC2-Benutzerhandbuch.

Änderung der Auftragsfunktion des Benutzers

An einem gewissen Punkt wechselt einer der Entwickler, Paulo Santos, die Auftragsfunktion und wird zum Manager. Als Manager wird Paulo Teil der Support-Benutzergruppe, sodass er Support-Anfragen für seine Entwickler eröffnen kann. Mateo verschiebt Paulo aus der Gruppe „Entwickler“ in die Gruppe „Support-Benutzer“. Infolge dieses Schritts ist seine Fähigkeit, mit Amazon EC2 Instances zu interagieren, beschränkt. Er kann keine Instances in Betrieb nehmen oder starten. Er kann auch keine vorhandenen Instances anhalten oder beenden, selbst wenn er der Benutzer war, der die Instance in Betrieb genommen oder gestartet hat. Er kann nur die Instances auflisten, die Example Corp-Benutzer gestartet haben.

Anwendungsfall für IAM mit Amazon S3

Unternehmen wie Example Corp würden in der Regel den IAM mit Amazon S3 verwenden. John hat einen Amazon S3-Bucket mit den Namen aws-s3-bucket für das Unternehmen erstellt.

Erstellen von anderen Benutzern und Gruppen

Als Mitarbeiter müssen Zhang Wei und Mary Major jeweils in der Lage sein, ihre eigenen Daten im Unternehmens-Buckets zu erstellen. Sie müssen auch freigegebene Daten, an denen alle Entwickler arbeiten, lesen und schreiben können. Um dies zu ermöglichen, ordnet Mateo die Daten logisch in

aws-s3-bucket, wobei er ein Amazon-S3-Schlüsselpräfixschema wie in der folgenden Abbildung gezeigt verwendet.

```
/aws-s3-bucket
    /home
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo unterteilt den `/aws-s3-bucket` in eine Reihe von Stammverzeichnissen für jeden Mitarbeiter sowie in einen gemeinsam genutzten Bereich für Gruppen aus Entwicklern und Managern.

Mateo erstellt nun eine Reihe von Richtlinien, um Berechtigungen für Benutzer und Gruppen zuzuweisen:

- Startverzeichniszugriff für Zhang – Mateo fügt eine Richtlinie an Wei an, mit der er beliebige Objekte mit dem Amazon S3-Schlüsselpräfix `/aws-s3-bucket/home/zhang/` lesen, schreiben und auflisten kann
- Startverzeichniszugriff für Major – Mateo fügt eine Richtlinie an Mary an, mit der sie beliebige Objekte mit dem Amazon S3-Schlüsselpräfix `/aws-s3-bucket/home/major/` lesen, schreiben und auflisten kann
- Freigegebene Verzeichniszugriff für die Entwicklergruppe – Mateo fügt eine Richtlinie an die Gruppe an, mit der Entwickler beliebige Objekte in lesen, schreiben und auflisten können. `/aws-s3-bucket/share/developers/`
- Freigegebener Verzeichniszugriff für die Managergruppe – Mateo fügt eine Richtlinie an die Gruppe an, mit die Manager Objekte in `/aws-s3-bucket/share/managers/` lesen, schreiben und auflisten können.

> **Note**
>
> Amazon S3 erteilt einem Benutzer, der einen Bucket oder ein Objekt erstellt, nicht automatisch die Berechtigung zum Ausführen anderer Aktionen auf diesem Bucket oder Objekt. Aus diesem Grund müssen Sie in Ihren IAM-Richtlinien die Benutzer zur Verwendung der Amazon S3-Ressourcen, die sie erstellen, explizit berechtigen.

Beispiele für diese Richtlinien finden Sie unter Zugriffskontrolle im Benutzerhandbuch für Amazon Simple Storage Service. Weitere Informationen über das Auswerten von Richtlinien zur Laufzeit finden Sie unter Auswertungslogik für Richtlinien.

Änderung der Auftragsfunktion des Benutzers

An einem gewissen Punkt wechselt einer der Entwickler, Zhang Wei, die Funktion und wird zum Manager. Wir gehen davon aus, dass er keinen Zugriff mehr auf die Dokumente im `share/developers`-Verzeichnis benötigt. Mateo, als Admin, verschiebt Wei in die `Managers`-Benutzergruppe und nimmt ihn aus der `Developers`-Benutzergruppe heraus. Mit dieser einfachen Neuzuweisung erhält Wei automatisch alle Berechtigungen der `Managers`-Benutzergruppe, kann aber nicht mehr auf Daten im `share/developers`-Verzeichnis zugreifen.

Integration in ein Drittunternehmen

Organisationen arbeiten häufig mit Partnerunternehmen, Beratern und Auftragnehmern. Example Corp hat eine Partner namens Widget Company und eine Mitarbeiterin dieses Unternehmens mit dem Namen Shirley Rodriguez muss Daten in einen Bucket platzieren, damit Example Corp sie nutzen kann. Nikki erstellt eine Benutzergruppe mit dem Namen WidgetCo`Shirley`und fügt Shirley der WidgetCo Benutzergruppe hinzu. Nikki erstellt auch einen speziellen Bucket namens aws-s3-bucket1, den Shirley verwenden kann.

Nikki aktualisiert vorhandene Richtlinien oder fügt neue hinzu, um den Partner Widget Company zu berücksichtigen. Nikki kann beispielsweise eine neue Richtlinie erstellen, die Mitgliedern der WidgetCo Benutzergruppe die Möglichkeit verweigert, andere Aktionen als das Schreiben zu verwenden. Diese Richtlinie wäre nur erforderlich, wenn es eine breite Richtlinie gibt, die allen Benutzern Zugriff auf eine große Menge von Amazon S3-Aktionen bietet.

IAM-Tutorials

In den folgenden Tutorials end-to-end werden vollständige Verfahren für allgemeine Aufgaben für AWS Identity and Access Management (IAM) vorgestellt. Sie sind für Laborumgebungen mit fiktiven Unternehmensnamen, Benutzernamen usw. vorgesehen. In den Tutorials finden Sie allgemeine Anleitungen. Ohne die sorgfältige Prüfung und Anpassung an die besonderen Gegebenheiten der Umgebung Ihrer Organisation sind sie nicht zur direkten Verwendung in einer Produktionsumgebung bestimmt.

Tutorials

- IAM-Tutorial: Gewähren von Zugriff auf die Fakturierungskonsole
- Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen
- IAM-Anleitung: Erstellen und Anfügen Ihrer ersten vom Kunden verwalteten Richtlinie
- IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren
- IAM-Tutorial: Zulassen, dass Ihre Benutzer ihre eigenen Anmeldeinformationen und MFA-Einstellungen konfigurieren können

IAM-Tutorial: Gewähren von Zugriff auf die Fakturierungskonsole

Der AWS-Konto Besitzer (Root-Benutzer des AWS-Kontos) kann IAM-Benutzern und -Rollen Zugriff auf die AWS Billing and Cost Management Daten für ihre Benutzer gewähren. AWS-Konto Die Anweisungen in diesem Tutorial helfen Ihnen, ein vorab getestetes Szenario einzurichten. Mit diesem Szenario können Sie praktische Erfahrungen in der Konfiguration von Fakturierungsberechtigungen sammeln, ohne dass dies Auswirkungen auf Ihr AWS -Hauptproduktionskonto hat.

Voraussetzungen

Treffen Sie die folgenden Vorbereitungen, bevor Sie die Schritte in diesem Tutorial ausführen:

- Erstellen Sie einen Test AWS-Konto.
- Melden Sie sich bei Ihrem Test AWS-Konto als Root-Benutzer an.
- Notieren Sie sich die AWS-Konto Nummer Ihres Testkontos, damit Sie es im Tutorial verwenden können. In diesem Tutorial verwenden wir die Beispielkontonummer 111122223333. Wenn in einem Schritt diese Kontonummer verwendet wird, ersetzen Sie sie durch Ihre Testkontonummer.

Schritt 1: Aktivieren Sie den IAM-Zugriff auf die Rechnungsinformationen in Ihrem Testkonto AWS

In diesem Szenario melden Sie sich bei Ihrem Test AWS-Konto als Root-Benutzer an, um IAM Zugriff auf die Rechnungsinformationen zu gewähren. Wenn Sie IAM Zugriff auf Rechnungsinformationen gewähren, können IAM-Benutzer und -Rollen auf die Konsole zugreifen. AWS Billing and Cost Management Diese Einstellung gewährt IAM-Benutzern und -Rollen nicht die erforderlichen Berechtigungen für diese Konsolenseiten, sondern ermöglicht den Zugriff für IAM-Benutzer oder -Rollen, die über die erforderlichen IAM-Richtlinien verfügen. Wenn IAM-Benutzern oder -Rollen bereits Richtlinien zugeordnet sind, diese Einstellung jedoch nicht aktiviert ist, sind die von diesen Richtlinien gewährten Berechtigungen nicht wirksam.

> **Note**
>
> AWS-Konten wurden erstellt, indem AWS Organizations der IAM-Zugriff auf Rechnungsinformationen standardmäßig aktiviert ist.

Schritt 2: Erstellen von Testbenutzern und -Gruppen

In diesem Szenario gewähren Sie IAM-Benutzern Zugriff auf die Fakturierungskonsole und erstellen zwei Benutzer:

- Pat Candella

  Pat ist Mitglied der Finanzabteilung und arbeitet mit Fakturierungen und Zahlungen. Pat benötigt vollen Zugriff auf die Rechnungsinformationen in Ihrem AWS-Konto.

- Terry Whitlock

  Terry ist Teil Ihrer IT-Supportabteilung. In den meisten Fällen benötigt Terry keinen Zugriff auf die Fakturierungskonsole, manchmal benötigt er jedoch Zugriff, um Fragen für Mitarbeiter in der Finanzabteilung zu beantworten.

Schritt 3: Erstellen einer Rolle, um Zugriff auf die AWS Billing -Konsole zu gewähren

Eine IAM-Rolle ist eine IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. Eine IAM-Rolle ähnelt einem IAM-Benutzer, da es sich um eine AWS -Identität mit Berechtigungsrichtlinien handelt, die bestimmen, was die Identität in AWS ausführen kann und was nicht. Eine Rolle ist jedoch nicht einer einzigen Person zugeordnet, sondern kann von allen Personen angenommen werden, die diese Rolle benötigen. Einer Rolle sind außerdem keine standardmäßigen, langfristigen Anmeldeinformationen, wie z. B. Passwörter

oder Zugriffsschlüssel, zugeordnet. Wenn Sie eine Rolle übernehmen, erhalten Sie stattdessen temporäre Anmeldeinformationen für Ihre Rollensitzung. Sie können Rollen verwenden, um den Zugriff an Benutzer, Anwendungen oder Dienste zu delegieren, die normalerweise keinen Zugriff auf Ihre AWS Ressourcen haben. In diesem Szenario erstellen Sie eine Rolle, die Terry Whitlock annehmen kann, um auf die Fakturierungskonsole zuzugreifen.

Schritt 4: Testen des Zugriffs auf die Konsole

Nachdem Sie diese zentralen Aufgaben abgeschlossen haben, können Sie die Richtlinie testen. Mit einem Test stellen Sie sicher, dass die Richtlinie wie gewünscht funktioniert. Durch das Testen des Zugriffs jedes Benutzers können Sie die Benutzererfahrungen vergleichen.

Voraussetzungen

Treffen Sie die folgenden Vorbereitungen, bevor Sie die Schritte in diesem Tutorial ausführen:

- Erstellen Sie einen Test AWS-Konto.
- Melden Sie sich bei Ihrem Test AWS-Konto als Root-Benutzer an.
- Notieren Sie sich die AWS-Konto Nummer Ihres Testkontos, damit Sie es im Tutorial verwenden können. In diesem Tutorial verwenden wir die Beispielkontonummer 111122223333. Wenn in einem Schritt diese Kontonummer verwendet wird, ersetzen Sie sie durch Ihre Testkontonummer.

Schritt 1: Aktivieren Sie den IAM-Zugriff auf die Rechnungsinformationen in Ihrem Testkonto AWS

In diesem Szenario melden Sie sich bei Ihrem Test AWS-Konto als Root-Benutzer an, um IAM Zugriff auf die Rechnungsinformationen zu gewähren. Wenn Sie Zugriff auf Rechnungsinformationen gewähren, können IAM-Benutzer und -Rollen auf die AWS Billing and Cost Management Konsole zugreifen. Diese Einstellung gewährt IAM-Benutzern und -Rollen nicht die erforderlichen Berechtigungen für diese Konsolenseiten, sondern ermöglicht nur den Zugriff für IAM-Benutzer oder -Rollen, die über die erforderlichen IAM-Richtlinien verfügen.

> **Note**
>
> AWS-Konten wurden erstellt, indem AWS Organizations der IAM-Zugriff auf Rechnungsinformationen standardmäßig aktiviert ist.

So aktivieren Sie den IAM-Benutzer- und -Rollenzugriff auf die Billing and Cost Management-Konsole

1. Melden Sie sich AWS Management Console mit Ihren Root-Benutzeranmeldedaten (insbesondere der E-Mail-Adresse und dem Passwort, mit denen Sie Ihr AWS Konto erstellt haben) bei der an.
2. Wählen Sie in der Navigationsleiste Ihren Kontonamen und dann Konto aus.
3. Scrollen Sie auf der Seite nach unten, bis Sie den Abschnitt IAM-Benutzer- und Rollenzugriff auf Abrechnungsinformationen finden, und wählen Sie dann Bearbeiten aus.
4. Aktivieren Sie das Kontrollkästchen Activate IAM Access (-Zugriff aktivieren), um den Zugriff auf die Billing and Cost Management-Seiten zu aktivieren.
5. Wählen Sie Update (Aktualisieren).

   Auf der Seite wird die Meldung angezeigt, dass der Zugriff des IAM-Benutzers/der IAM-Rolle auf die Abrechnungsinformationen aktiviert ist.

Im nächsten Schritt dieses Tutorials fügen Sie IAM-Richtlinien hinzu, um den Zugriff auf bestimmte Fakturierungsfunktionen zu gewähren oder zu verweigern.

Schritt 2: Erstellen von Testbenutzern und -Gruppen

Für Ihr AWS Testkonto sind außer dem Root-Benutzer keine Identitäten definiert. Um Zugriff auf Abrechnungsinformationen zu ermöglichen, stellen wir zusätzliche Identitäten bereit, denen wir die Berechtigung zum Zugriff auf Abrechnungsinformationen gewähren können.

Erstellen von Testbenutzern und -Gruppen

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

   > **Note**
   >
   > Als Root-Benutzer können Sie sich nicht auf der Seite Sign in as IAM user (Als IAM-Benutzer anmelden) anmelden. Wenn Ihnen die Seite Sign in as IAM user (Als IAM-Benutzer anmelden) angezeigt wird, wählen Sie die Option Sign in using root user email (Mit Root-Benutzer-E-Mail anmelden) unten auf der Seite. Hilfe bei der Anmeldung als

Root-Benutzer finden Sie im Benutzerhandbuch unter AWS Management Console Als AWS-Anmeldung Root-Benutzer anmelden.

2. Wählen Sie im Navigationsbereich Users (Benutzer) und dann Add users (Benutzer hinzufügen) aus.

> **Note**
>
> Wenn Sie IAM Identity Center aktiviert haben, AWS Management Console wird eine Erinnerung angezeigt, dass es am besten ist, den Benutzerzugriff im IAM Identity Center zu verwalten. In diesem Tutorial erfahren die von uns erstellten IAM-Benutzer, wie der Zugriff auf Abrechnungsinformationen gewährt wird. Wenn Sie Benutzer im IAM Identity Center erstellt haben, weisen Sie den Berechtigungssatz Fakturierung diesen Benutzern oder Gruppen zu, indem Sie IAM Identity Center anstelle von IAM verwenden.

3. Geben Sie unter User Name (Benutzername) den Text **`pcandella`** ein. Namen dürfen keine Leerzeichen enthalten.
4. Aktivieren Sie das Auswahlfeld neben Benutzerzugriff auf bereitstellen AWS Management Console— optional und wählen Sie dann „Möchten Sie einen IAM-Benutzer erstellen“ aus.
5. Wählen Sie unter Console password (Konsolenpasswort) Autogenerated password (Automatisch generiertes Passwort).
6. Deaktivieren Sie das Kontrollkästchen neben Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen (empfohlen) und wählen Sie dann Weiter. Da es sich bei diesem IAM-Benutzer um einen Testbenutzer handelt, laden wir das Kennwort herunter, um es während des Verifizierungsverfahrens zu verwenden.
7. Wählen Sie auf der Seite Set permissions (Berechtigungen festlegen) unter Permissions options (Berechtigungsoptionen) die Option Add user to group (Benutzer zur Gruppe hinzufügen) aus. Wählen Sie dann unter User groups (Benutzergruppen) die Option Create group (Gruppe erstellen) aus.
8. Geben Sie auf der Seite Create user group (Benutzergruppe erstellen) im Feld User group name (Benutzergruppenname) **`BillingGroup`** ein. Wählen Sie dann unter Berechtigungsrichtlinien die Richtlinie Abrechnung für AWS verwaltete Jobfunktionen aus.
9. Wählen Sie Create user group (Benutzergruppe erstellen) aus, um zur Seite Set permissions (Berechtigungen festlegen) zurückzukehren.
10. Wählen Sie unter Benutzergruppen das Auswahlfeld des von Ihnen erstellten **`BillingGroup`** aus.

11. Wählen Sie Next (Weiter), um mit der Seite Review and create (Überprüfen und erstellen) fortzufahren.
12. Überprüfen Sie auf der Seite Überprüfen und erstellen die Liste der Benutzergruppenmitgliedschaften für den neuen Benutzer. Wenn Sie bereit sind, fortzufahren, wählen Sie Create user (Benutzer erstellen) aus.
13. Wählen Sie auf der Seite Passwort abrufen die Option CSV-Datei herunterladen aus, um eine CSV-Datei mit den Anmeldeinformationen des Benutzers (Verbindungs-URL, Benutzername und Passwort) zu speichern.

    Speichern Sie diese Datei, um sie als Referenz zu verwenden, wenn Sie sich AWS als dieser IAM-Benutzer anmelden
14. Zurück zur Benutzerliste auswählen
15. Wiederholen Sie diesen Vorgang mit den folgenden Änderungen, um den Benutzer für Terry Whitlock und eine Gruppe für Supportbenutzer zu erstellen.

    a. Geben Sie in Schritt 3 als Benutzername **`twhitlock`** ein.

    b. Geben Sie in Schritt 8 als Benutzergruppenname **`SupportGroup`** ein. Wählen Sie dann unter Berechtigungsrichtlinien die Funktionsrichtlinie für AWS verwaltete Jobs aus. SupportUser

Sie können die neuen IAM-Benutzer, -Gruppen und -Rollen in den Konsolenlisten überprüfen. Für jedes von Ihnen erstellte Element können Sie den Namen auswählen, um dessen Details anzuzeigen. Wenn Sie sich die Benutzerdetails ansehen, zeigt die Konsole die Abrechnung unter Berechtigungsrichtlinien für **`pcandella`** und SupportUserunter Berechtigungsrichtlinien für an. **`twhitlock`**

Weitere Informationen zur Verwendung von Richtlinien, um IAM-Benutzern Zugriff auf AWS Billing and Cost Management -Funktionen zu gewähren, finden Sie unter Verwenden von identitätsbasierten Richtlinien (IAM-Richtlinien) für AWS Billing im AWS Billing -Benutzerhandbuch.

Schritt 3: Erstellen einer Rolle, um Zugriff auf die AWS Billing -Konsole zu gewähren

Sie können eine Rolle verwenden, um IAM-Benutzern Zugriff auf die Fakturierungskonsole zu gewähren. Rollen stellen temporäre Anmeldeinformationen bereit, die Benutzer bei Bedarf übernehmen können. In diesem Tutorial muss der Benutzer **`twhitlock`** auf

Abrechnungsinformationen zugreifen können, wenn er aufgrund einer Supportanfrage der Finanzabteilung ein Problem untersuchen muss.

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

   > **Note**
   >
   > Als Root-Benutzer können Sie sich nicht auf der Seite Sign in as IAM user (Als IAM-Benutzer anmelden) anmelden. Wenn Ihnen die Seite Sign in as IAM user (Als IAM-Benutzer anmelden) angezeigt wird, wählen Sie die Option Sign in using root user email (Mit Root-Benutzer-E-Mail anmelden) unten auf der Seite. Hilfe bei der Anmeldung als Root-Benutzer finden Sie im Benutzerhandbuch unter AWS Management Console Als AWS-Anmeldung Root-Benutzer anmelden.

2. Wählen Sie im Navigationsbereich Benutzer und dann den **`twhitlock`**-Benutzer aus, um die Benutzerdetails anzuzeigen. Kopieren Sie den ARN für den **`twhitlock`**-Benutzer in die Zwischenablage.
3. Wählen Sie im Navigationsbereich Rollen und dann Rolle erstellen aus.
4. Wählen Sie auf der Seite Vertrauenswürdige Entität auswählen die Option Benutzerdefinierte Vertrauensrichtlinie aus und füllen Sie dann unter Anweisung bearbeiten die folgenden Elemente aus:

   - Aktionen für STS hinzufügen — Vergewissern Sie AssumeRolesich, dass diese Option ausgewählt ist.
   - Prinzipal hinzufügen – Wählen Sie Hinzufügen, um das Dialogfeld Prinzipal hinzufügen anzuzeigen. Wählen Sie unter Prinzipal-Typ die Option IAM-Benutzer aus und fügen Sie dann unter ARN den ARN für den Twhitlock-Benutzer ein, den Sie in Schritt 16 in die Zwischenablage kopiert haben. Wählen Sie anschließend Prinzipal hinzufügen aus.

5. Wählen Sie Weiter, um zur Seite Berechtigungen hinzufügen zu gelangen.
6. Geben Sie im Filterfeld unter Berechtigungsrichtlinien die Richtlinie Abrechnung für AWS verwaltete Aufgaben ein **`Billing`** und wählen Sie sie aus.
7. Wählen Sie Weiter aus, um zur Seite Benennen, Überprüfen und Erstellen zu gelangen. Geben Sie unter Rollenname **`TempBillingAccess`** ein und wählen Sie dann Rolle erstellen aus.

Sie werden darüber informiert, dass die Rolle erstellt wurde. Zeigen Sie die Rolle an, um die Details zur Rolle anzuzeigen. Beachten Sie im Abschnitt Zusammenfassung die folgenden Informationen:

- Die maximale Sitzungsdauer beträgt standardmäßig 1 Stunde. Nach dieser Zeit erhält der Benutzer, der die Rolle übernommen hat, wieder die Berechtigungen seines Basiskontos. Wenn der Benutzer die Rollenberechtigungen weiterhin nutzen möchte, muss er die Rolle erneut wechseln. Sie können die Rolle bearbeiten, um die maximale Dauer zu erhöhen. Die längste mögliche Sitzungsdauer beträgt 12 Stunden.
- Link zum Rollenwechsel in der Konsole. Sie können den Link kopieren, um ihn direkt den Benutzern bereitzustellen, die Sie als Prinzipale in der Vertrauensrichtlinie hinzufügen. Sie können die Vertrauensrichtlinie auf der Registerkarte Vertrauensstellungen anzeigen und bearbeiten.

Schritt 4: Testen des Zugriffs auf die Konsole

Wir empfehlen Ihnen, den Zugriff zu testen, indem Sie sich als die Testbenutzer anmelden, sodass Sie das potenzielle Erlebnis Ihrer Benutzer nachvollziehen können. Mit den folgenden Schritten melden Sie sich bei beiden Testkonten an, um die unterschiedlichen Zugriffsberechtigungen zu testen.

So testen Sie den Fakturierungszugriff, indem Sie sich bei beiden Testbenutzern anmelden

1. Verwenden Sie Ihre AWS Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der IAM-Konsole anzumelden.

   > **Note**
   >
   > Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich Ihren IAM-Benutzernamen und Ihre Kontoinformationen zu merken. Wenn Sie sich zuvor als anderer Benutzer angemeldet haben, wählen Sie Melden Sie sich bei einem anderen Konto an Um zur Hauptanmeldeseite zurückzukehren. Von dort aus können Sie Ihre AWS Konto-ID oder Ihren Kontoalias eingeben, um zur IAM-Benutzer-Anmeldeseite für Ihr Konto weitergeleitet zu werden.

2. Melden Sie sich wie nachfolgend beschrieben bei jedem Benutzer an, um die unterschiedlichen Benutzererfahrungen zu testen.

Vollzugriff

a. Melden Sie sich AWS-Konto als Benutzer bei Ihnen an. **`pcandella`**
b. Wählen Sie in der Navigationsleiste pcandella @111122223333 und dann Fakturierungs-Dashboard aus.
c. Klicken Sie auf die Seiten und wählen Sie die verschiedenen Schaltflächen aus, um sicherzustellen, dass Sie über vollständige Änderungsberechtigungen verfügen.

Kein Zugriff

a. Melden Sie sich bei Ihnen AWS-Konto als Benutzer an**`twhitlock`**.
b. Wählen Sie in der Navigationsleiste twhitlock@111122223333 und dann Fakturierungs-Dashboard aus.
c. Es wird eine Meldung mit dem Hinweis Sie benötigen Berechtigungen angezeigt. Es sind keine Fakturierungsdaten sichtbar.

Rolle wechseln, um den Zugriff zu erhöhen

a. Melden Sie sich bei Ihnen AWS-Konto als Benutzer an**`twhitlock`**.
b. Wählen Sie in der Navigationsleiste twhitlock @111122223333 und dann Rolle wechseln aus.

   Die Seite Rolle wechseln wird geöffnet. Vervollständigen Sie die Angaben wie folgt:

   - Konto – 111122223333
   - Rolle – **`TempBillingAccess`**

   Wählen Sie Rolle wechseln

   Alternativ können Sie auch die unter Link zum Rollenwechsel in der Konsole bereitgestellte URL verwenden, um die Seite Rolle wechseln zu öffnen.
c. In der Konsole wird das AWS Billing Dashboard und in der Navigationsleiste TempBillingAccess@111122223333 angezeigt.

Übersicht

Sie haben nun die erforderlichen Schritte abgeschlossen, um IAM-Benutzern Zugriff auf die AWS Billing -Konsole zu gewähren. Dadurch haben Sie aus erster Hand gesehen, wie die Erfahrung Ihrer Benutzer mit der Fakturierungskonsole aussieht. Sie können nun nach Belieben mit der Implementierung dieser Logik in Ihre Produktionsumgebung fortfahren.

Zugehörige Ressourcen

Weitere Informationen aus dem Leitfaden AWS Billing IAM-Leitfaden finden Sie in folgenden verwandten Ressourcen:

- Zugriff auf die AWS Billing Konsole aktivieren
- AWS Beispiele für Abrechnungsrichtlinien
- Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für die Abrechnung AWS
- Migration der Zugriffskontrolle für AWS Billing

Weitere Informationen im IAM-Leitfaden finden Sie in folgenden verwandten Ressourcen:

- Verwaltete Richtlinien und eingebundene Richtlinien
- Steuern des IAM-Benutzerzugriffs auf die AWS Management Console
- Zuordnen einer Richtlinie zu einer IAM-Gruppe

Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen

In diesem Tutorial erfahren Sie, wie Sie mit einer Rolle den Zugriff auf Ressourcen in anderer Ihrer AWS-Konten namens Produktion und Entwicklung delegieren. Sie teilen Ressourcen in einem Konto mit Benutzern in einem anderen Konto. Indem Sie auf diese Weise den kontoübergreifenden Zugriff einrichten, müssen Sie keine einzelnen IAM-Benutzer in jedem Konto erstellen. Darüber hinaus müssen sich Benutzer nicht von einem Konto abmelden und bei einem anderen Konto anmelden, um auf Ressourcen in unterschiedlichen AWS-Konten zuzugreifen. Nachdem Sie die Rolle konfiguriert haben, erfahren Sie, wie Sie die Rolle über die AWS Management Console AWS CLI, und die API verwenden.

> **Note**
>
> IAM-Rollen und ressourcenbasierte Richtlinien delegieren den Zugriff auf Konten nur innerhalb einer einzelnen Partition. Nehmen wir zum Beispiel an, Sie haben ein Konto in US West (Nordkalifornien) in der Standardpartition `aws`. Sie haben auch ein Konto in China (Peking) in der `aws-cn`-Partition. Sie können keine ressourcenbasierte Amazon S3-Richtlinie in Ihrem Konto in China (Peking) verwenden, um Benutzern in Ihrem Standard-`aws`-Konto den Zugriff zu ermöglichen.

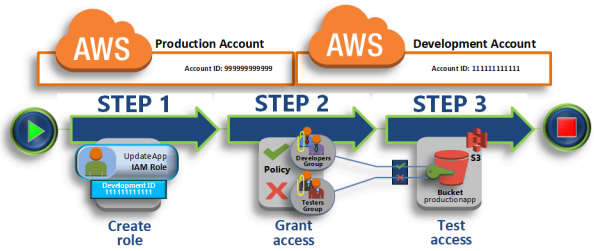
In diesem Tutorial verwaltet die Produktionsabrechnung Live-Anwendungen. Entwickler und Tester verwenden das Entwicklungskonto als eine Sandbox, um Anwendungen frei zu testen. In jedem Konto werden Anwendungsinformationen in Amazon-S3-Buckets gespeichert. Sie verwalten IAM-Benutzer im Entwicklungskonto, in dem Sie über zwei IAM-Gruppen verfügen: Entwickler und Tester. Die Benutzer in beiden Gruppen verfügen über Berechtigungen für die Nutzung des Development-Kontos und den Zugriff auf dessen Ressourcen. Von Zeit zu Zeit muss ein Entwickler die Live-Anwendungen in der Produktionsabrechnung aktualisieren. Die Entwickler speichern diese Anwendungen in einem Amazon-S3-Bucket mit dem Namen `productionapp`.

Am Ende dieses Tutorials ist Folgendes verfügbar:

- Benutzer im Entwicklungskonto (das vertrauenswürdige Konto), die eine bestimmte Rolle in der Produktionsabrechnung übernehmen dürfen.
- Eine Rolle in der Produktionsabrechnung (das vertrauende Konto), die auf einen bestimmten Amazon-S3-Bucket zugreifen darf.
- Ein `productionapp`-Bucket in der Produktionsabrechnung.

Entwickler können die Rolle in verwenden AWS Management Console , um auf den `productionapp` Bucket im Produktionskonto zuzugreifen. Sie können auf den Bucket auch mithilfe von API-Aufrufen zugreifen, die anhand von der Rolle bereitgestellten temporären Anmeldeinformationen authentifiziert werden. Tester können diese Rolle nicht verwenden.

Dieser Workflow umfasst drei grundlegende Schritte:



Erstellen einer Rolle im Produktionskonto

Zunächst verwenden Sie den, AWS Management Console um eine Vertrauensstellung zwischen dem Produktionskonto (ID-Nummer 9999999999) und dem Entwicklungskonto (ID-Nummer 111111111111) herzustellen. Sie erstellen UpdateAppzunächst eine IAM-Rolle mit dem Namen. Wenn Sie die Rolle erstellen, definieren Sie das Entwicklungskonto als vertrauenswürdige Entität und geben Sie eine Berechtigungsrichtlinie an, die den vertrauenswürdigen Benutzern die Aktualisierung des `productionapp`-Buckets gestattet.

Erteilen der Zugriffsberechtigung auf die Rolle

In diesem Abschnitt ändern Sie die IAM-Benutzergruppenrichtlinie so, dass Testern der Zugriff auf die `UpdateApp`-Rolle verweigert wird. Weil Tester in diesem Szenario PowerUser Zugriff haben und Sie die Nutzung der Rolle ausdrücklich verweigern müssen.

Testen des Zugriffs durch Rollenwechsel

Schließlich verwenden Sie als Entwickler die `UpdateApp`-Rolle zum Aktualisieren des `productionapp`-Buckets in der Produktionsabrechnung. Sie erfahren, wie Sie über die AWS Konsole, die und die API auf die AWS CLI Rolle zugreifen können.

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass Folgendes bereits vorhanden ist:

- Zwei separate AWS-Konten , die Sie verwenden können, eines für das Entwicklungskonto und eines für das Produktionskonto.
- Benutzer und Benutzergruppen im Entwicklungskonto, wie folgt erstellt und konfiguriert:

Benutzer	Benutzergruppe	Berechtigungen
David	Entwickler	Beide Benutzer können sich anmelden und das Konto AWS Management Console im Entwicklerkonto verwenden.
Jane	Tester	

- Sie benötigen keine Benutzer oder Benutzergruppen, die in der Produktionsabrechnung erstellt wurden.
- Ein Amazon-S3-Bucket, das in der Produktionsabrechnung erstellt wurde. In diesem Tutorial nennen wir dies `ProductionApp`, S3-Bucket-Namen sind jedoch global eindeutig, daher müssen Sie einen Bucket mit einem anderen Namen verwenden.

Erstellen einer Rolle im Produktionskonto

Sie können Benutzern von einem Standort AWS-Konto den Zugriff auf Ressourcen in einem anderen ermöglichen AWS-Konto. Erstellen Sie dazu eine Rolle, die definiert, wer darauf zugreifen kann und welche Berechtigungen Benutzern gewährt werden, die wechseln.

In diesem Schritt des Tutorials erstellen Sie die Rolle in der Produktionsabrechnung und geben das Entwicklungskonto als vertrauenswürdige Entität an. Außerdem beschränken Sie die Berechtigungen der Rolle auf den Lese- und Schreibzugriff auf den Bucket `productionapp`. Jeder Benutzer, dem die Berechtigung zur Verwendung der Rolle erteilt wird, kann auf dem Bucket `productionapp` Lese- und Schreibvorgänge ausführen.

Bevor Sie eine Rolle erstellen können, benötigen Sie die Konto-ID der Entwicklung AWS-Konto. Jedem AWS-Konto ist eine eindeutige Konto-ID-ID zugewiesen.

Um die AWS-Konto Entwicklungs-ID zu erhalten

1. Melden Sie sich AWS Management Console als Administrator des Entwicklerkontos an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich auf Support und dann auf Support Center. Ihre aktuell angemeldete 12-stellige Kontonummer (ID) wird im Bereich Support-Center-Navigationsbereich. Für dieses Szenario können Sie die Konto-ID 111111111111 für das Entwicklungskonto verwenden. Sie sollten jedoch eine gültige Konto-ID verwenden, wenn Sie das Szenario in Ihrer Testumgebung rekonstruieren.

So erstellen Sie eine Rolle im Produktionskonto, die vom Entwicklungs-Konto verwendet werden kann

1. Melden Sie sich AWS Management Console als Administrator des Produktionskontos an und öffnen Sie die IAM-Konsole.
2. Bevor Sie die Rolle erstellen, richten Sie die verwaltete Richtlinie ein, die die von der Rolle benötigten Berechtigungen definiert. Diese Richtlinie fügen Sie zu einem späteren Zeitpunkt der Rolle an.

   Sie möchten Lese- und Schreibberechtigungen für den Bucket `productionapp` erteilen. Obwohl AWS einige verwaltete Amazon S3 S3-Richtlinien bereitgestellt werden, gibt es keine, die Lese- und Schreibzugriff auf einen einzelnen Amazon S3 S3-Bucket bietet. Sie können aber Ihre eigene Richtlinie erstellen.

   Wählen Sie im Navigationsbereich Policies (Richtlinien) und dann Create policy (Richtlinie erstellen) aus.
3. Wählen Sie die Registerkarte JSON aus und kopieren Sie den Text aus dem folgenden JSON-Richtliniendokument. Fügen Sie den Text in das Textfeld JSON ein und ersetzen Sie dabei den Ressourcen-ARN (`arn:aws:s3:::productionapp`) durch den passenden ARN für Ihren Amazon-S3-Bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
```

```
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Die Aktion `ListAllMyBuckets` gewährt die Berechtigung zum Auflisten aller Buckets, die dem authentifizierten Sender der Anforderung gehören. Die Berechtigung `ListBucket` ermöglicht den Benutzern, Objekte im Bucket `productionapp` anzuzeigen. Die Berechtigungen `GetObject`, `PutObject` und `DeleteObject` ermöglichen den Benutzern, Inhalte im Bucket `productionapp` anzuzeigen, zu aktualisieren und zu löschen.

4. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

5. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) als Richtliniennamen **`read-write-app-bucket`** ein. Überprüfen Sie die von ihrer Richtlinie erteilten Berechtigungen und wählen Sie dann zum Speichern Ihrer Arbeit Create policy (Richtlinie erstellen) aus.

   Die neue Richtlinie wird in der Liste der verwalteten Richtlinien angezeigt.

6. Wählen Sie im Navigationsbereich Rollen und dann Rolle erstellen.
7. Wählen Sie den Rollentyp AWS-Konto.
8. Geben Sie für Account ID (Konto-ID), die Entwicklungskonto-ID an.

   In diesem Tutorial wird für das Entwicklungs-Konto die Beispiel-Konto-ID **111111111111** verwendet. Verwenden Sie eine gültige Konto-ID. Wenn Sie eine ungültige Konto-ID verwenden, wie z. B. **111111111111**, lässt Sie IAM keine neue Rolle erstellen.

Zu diesem Zeitpunkt benötigen Sie jedoch noch keine externe ID und müssen keine Multi-Faktor-Authentifizierung (MFA) von den Benutzern verlangen, um die Rolle zu übernehmen. Wählen Sie daher diese Option nicht. Weitere Informationen finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

9. Wählen Sie Next: Permissions (Weiter: Berechtigungen) aus, um die mit der Rolle verknüpften Berechtigungen einzurichten.
10. Markieren Sie das Kontrollkästchen neben der vorher erstellten Richtlinie.

> **Tipp**
>
> Klicken Sie bei Filter auf Customer managed (Benutzerdefiniert), um nur die von Ihnen erstellten Richtlinien anzuzeigen. Dadurch werden die von AWS erstellten Richtlinien ausgeblendet und Ihnen wird das Auffinden der gesuchten Richtlinie erleichtert.

Wählen Sie anschließend Weiter.

11. (Optional) Fügen Sie der Rolle Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
12. (Optional) Geben Sie unter Role description (Rollenbeschreibung) eine Beschreibung für die neue Rolle ein.
13. Nachdem Sie die Rolle überprüft haben, klicken Sie auf Create role (Rolle erstellen).

Die Rolle `UpdateApp` erscheint in der Liste der Rollen.

Jetzt müssen Sie den Amazon-Ressourcennamen (ARN) ermitteln. Dabei handelt es sich um eine eindeutige Kennung der Rolle. Wenn Sie die Richtlinie der Entwickler- und Tester-Benutzergruppe ändern, geben Sie den ARN der Rolle an, um Berechtigungen zu erteilen oder zu verweigern.

Um den ARN zu erhalten für UpdateApp

1. Wählen Sie im Navigationsbereich der IAM Console Roles.
2. Wählen Sie in der Rollenliste die Rolle `UpdateApp`.
3. Im Abschnitt Summary (Übersicht) im Detailbereich kopieren Sie den Wert Role ARN (Rollen-ARN).

Das Produktionskonto hat eine Konto-ID von 999999999999, die ARN der Rolle ist also `arn:aws:iam::999999999999:role/UpdateApp`. Stellen Sie sicher, dass Sie die echte AWS-Konto ID für das Produktionskonto angeben.

An diesem Punkt haben Sie eine Vertrauensstellung zwischen den Produktions- und Entwicklungskonten eingerichtet. Dazu haben Sie eine Rolle in der Produktionsabrechnung erstellt, die das Entwicklungskonto als vertrauenswürdigen Auftraggeber identifiziert. Sie haben auch definiert, was die Benutzer tun dürfen, die in die `UpdateApp`-Rolle wechseln.

Ändern Sie dann die Berechtigungen für die Benutzergruppen.

Erteilen der Zugriffsberechtigung auf die Rolle

An diesem Punkt verfügen sowohl die Tester als auch die Entwickler über Berechtigungen, um Anwendungen im Entwicklungskonto uneingeschränkt zu testen. Nachstehend werden die erforderlichen Schritte zum Hinzufügen von Berechtigungen beschrieben, um zur Rolle zu wechseln.

Um die Benutzergruppe „Entwickler“ so zu ändern, dass sie zur UpdateApp Rolle wechseln können

1. Melden Sie sich als Administrator beim Entwicklungskonto an und öffnen Sie die IAM-Konsole.
2. Wählen Sie Groups (Gruppen) und dann Developers (Entwickler).
3. Wählen Sie die Registerkarte Berechtigungen, wählen Sie Berechtigungen hinzufügen und wählen Sie dann Inline-Richtlinie erstellen.
4. Wählen Sie den Tab JSON.
5. Fügen Sie die folgende Richtlinienanweisung hinzu, um die Aktion `AssumeRole` in der Rolle `UpdateApp` im Production-Konto zu ermöglichen. Stellen Sie sicher, dass Sie *`PRODUCTION-ACCOUNT-ID`* im `Resource` Element auf die tatsächliche AWS-Konto ID des Produktionskontos ändern.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
  }
}
```

Die Aktion `Allow` gewährt der Entwickler-Gruppe explizit den Zugriff auf die Rolle `UpdateApp` im Production-Konto. Alle Entwickler können problemlos auf die Rolle zugreifen.

6. Wählen Sie Richtlinie prüfen.
7. Geben Sie einen Namen ein, z. B. **`allow-assume-S3-role-in-production`**.
8. Wählen Sie Richtlinie erstellen aus.

In den meisten Umgebungen ist folgende Vorgehensweise wahrscheinlich nicht erforderlich. Wenn Sie jedoch PowerUserAccess Berechtigungen verwenden, können einige Gruppen möglicherweise bereits die Rollen wechseln. Das folgende Verfahren zeigt, wie Sie der Gruppe Tester die Berechtigung `"Deny"` hinzufügen, um sicherzustellen, dass sie die Rolle nicht übernehmen können. Wenn dieses Verfahren in Ihrer Umgebung nicht benötigt wird, empfehlen wir Ihnen, es nicht hinzuzufügen. `"Deny"`-Berechtigungen machen die Verwaltung und das Verständnis der gesamten Berechtigungsstruktur komplizierter. Verwenden Sie `"Deny"`-Berechtigungen nur, wenn es keine bessere Option gibt.

Um die Benutzergruppe der Tester so zu ändern, dass sie die Berechtigung zur Übernahme der **`UpdateApp`**-Rolle nicht erhält

1. Wählen Sie Groups (Gruppen) und dann Testers (Tester).
2. Wählen Sie die Registerkarte Berechtigungen, wählen Sie Berechtigungen hinzufügen und wählen Sie dann Inline-Richtlinie erstellen.
3. Wählen Sie den Tab JSON.
4. Fügen Sie die folgende Richtlinienanweisung hinzu, um die Aktion `AssumeRole` in der Rolle `UpdateApp` zu verweigern. Stellen Sie sicher, dass Sie *`PRODUCTION-ACCOUNT-ID`* im `Resource` Element auf die tatsächliche AWS-Konto ID des Produktionskontos ändern.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
  }
}
```

Die Aktion `Deny` verweigert der Tester-Gruppe explizit den Zugriff auf die Rolle `UpdateApp` im Production-Konto. Allen Testern, die versuchen, auf die Rolle zuzugreifen, wird der Zugriff verweigert.

5. Wählen Sie Richtlinie prüfen.
6. Geben Sie einen Namen wie **`deny-assume-S3-role-in-production`** ein.
7. Wählen Sie Richtlinie erstellen aus.

Die Entwickler-Gruppe verfügt jetzt über Berechtigungen für die Verwendung der Rolle `UpdateApp` im Production-Konto. Der Tester-Gruppe wird der Zugriff auf die Rolle `UpdateApp` verweigert.

Als Nächstes erfahren Sie, wie der Entwickler David auf den `productionapp`-Bucket in der Produktionsabrechnung zugreifen kann. David kann über die, die oder die API auf den AWS Management Console Bucket zugreifen. AWS CLI AWS

Testen des Zugriffs durch Rollenwechsel

Nachdem Sie die ersten beiden Schritte dieses Tutorials abgeschlossen haben, verfügen Sie über eine Rolle, die den Zugriff auf eine Ressource in der Produktionsabrechnung gewährt. Außerdem verfügen Sie über eine Gruppe im Entwicklungskonto, deren Benutzer diese Rolle verwenden dürfen. In diesem Schritt wird erläutert, wie Sie den Wechsel von der AWS Management Console, der und der AWS CLI AWS API zu dieser Rolle testen können.

> **Important**
>
> Um zu einer anderen Rolle zu wechseln, müssen Sie sich als IAM-Benutzer oder Verbundbenutzer angemeldet haben. Wenn Sie eine Amazon EC2-Instance starten, um eine Anwendung auszuführen, kann die Anwendung außerdem über ihr Instance-Profil eine Rolle übernehmen. Sie können nicht zu einer Rolle wechseln, wenn Sie sich als Root-Benutzer des AWS-Kontos anmelden.

Wechseln der Rollen (Konsole)

Wenn David in der Produktionsumgebung von arbeiten muss AWS Management Console, kann er dies mithilfe von Switch Role tun. Nach Angabe der Konto-ID oder des Alias und des Rollennamens erfolgt sofort der Wechsel zu den von der Rolle erteilten Berechtigungen. Er kann dann die Konsole zum Arbeiten mit dem `productionapp` Bucket verwenden, allerdings hat er keinen Zugriff auf alle

anderen Ressourcen in Produktion. Während David die Rolle verwendet, kann er auch nicht seine Power-User-Privilegien im Entwicklungskonto nutzen. Dies liegt daran, dass jeweils nur eine Gruppe von Berechtigungen wirksam sein kann.

> **Important**
>
> Das Wechseln der Rollen mithilfe von funktioniert AWS Management Console nur mit Konten, für die kein Konto erforderlich ist`ExternalId`. Nehmen wir an, dass Sie Dritten Zugriff auf Ihr Konto gewähren und eine `ExternalId` in einem `Condition`-Element in Ihrer Berechtigungsrichtlinie benötigen. In diesem Fall kann der Drittanbieter nur über die AWS API oder ein Befehlszeilentool auf Ihr Konto zugreifen. Der Dritte kann nicht die Konsole verwenden, da sie keine `ExternalId` bereitstellt. Weitere Informationen zu diesem Szenario finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren und So aktivieren Sie den kontoübergreifenden Zugriff AWS Management Console im AWS Sicherheitsblog.

IAM bietet zwei Möglichkeiten, mit denen David die Seite Switch Role (Rolle wechseln) aufrufen kann:

- David erhält einen Link von seinem Administrator, der auf eine vordefinierte Konfiguration zum Wechseln der Rolle verweist. Der Link wird dem Administrator auf der letzten Seite des Assistenten Create role (Rolle wechseln) oder auf der Seite Role Summary (Rollenübersicht) für kontoübergreifende Rollen bereitgestellt. Wenn David diesem Link folgt, wird er zur Seite Switch Role (Rolle wechseln) weitergeleitet, in der die Felder Account ID (Konto-ID) und Role name (Rollenname) bereits ausgefüllt sind. David muss lediglich Switch Roles (Rollen wechseln) auswählen.
- Der Administrator sendet nicht den Link mit der E-Mail, sondern die Werte für die Konto-ID-Nummer und den Rollennamen. Um die Rollen zu wechseln, muss David die -Werte manuell eingeben. Dies wird in der folgenden Anleitung veranschaulicht.

So übernehmen Sie eine Rolle

1. David meldet sich AWS Management Console mit seinem normalen Benutzer in der Benutzergruppe Development an.
2. Er wählt den Link, den ihm der Administrator per E-Mail zugeschickt hat. Dadurch wird David auf die Seite Switch Role (Rolle wechseln) weitergeleitet, in der die Angaben zu Konto-ID und Rollenname bereits ausgefüllt sind.

–oder–

David wählt seinen Namen (im Identity-Menü) auf der Navigationsleiste aus und wählt dann Switch Roles (Rollen wechseln).

Wenn dies das erste Mal ist, dass David versucht, auf die Seite "Switch Role" zuzugreifen, landet er zunächst auf einer initialen Switch Role (Rolle wechseln)-Seite. Auf dieser Seite finden Sie weitere Informationen darüber, wie durch das Wechseln von Rollen die Benutzer befähigt werden, Ressourcen in verschiedenen AWS-Konten zu verwalten. David muss auf dieser Seite auf die Schaltfläche Switch Role (Rolle wechseln) klicken, um den Rest des beschriebenen Verfahrens abzuschließen.

3. Als Nächstes muss David manuell die ID-Nummer des produktiven Kontos (`999999999999`) und den Rollennamen (`UpdateApp`) eingeben, um auf die Rolle zugreifen zu können.

   David möchte außerdem überwachen, welche Rollen und zugehörigen Berechtigungen derzeit in IAM aktiv sind. Zum Nachverfolgen dieser Informationen gibt er `PRODUCTION` in das Textfeld Display Name (Anzeigename) ein, wählt die Option für Rot und dann Switch Role (Rolle wechseln).

4. David kann nun die Amazon S3-Konsole verwenden, um mit dem Amazon S3-Bucket oder einer anderen Ressource zu arbeiten, für die die `UpdateApp`-Rolle Berechtigungen hat.

5. Sobald dies erledigt ist, kann David zu seinen ursprünglichen Berechtigungen zurückkehren. Hierzu muss er den Anzeigenamen der PRODUCTION-Rolle aus der Navigationsleiste wählen und dann Back to David @ 111111111111 (Zurück zu David @ 111111111111).

6. Wenn David das nächste Mal die Rolle wechseln möchte und das Identitätsmenü in der Navigationsleiste aufruft, sieht er immer noch den Eintrag PRODUKTION vom letzten Mal. Durch Klicken auf diesen Eintrag kann er die Rolle sofort wechseln, ohne die Konto-ID und den Rollennamen erneut angeben zu müssen.

Wechseln der Rolle (AWS CLI)

Wenn David in der Produktionsumgebung mit der Befehlszeile arbeiten muss, ist dies über die AWS CLI möglich. Er führt den Befehl `aws sts assume-role` aus und übergibt den ARN der Rolle, um temporäre Sicherheitsanmeldeinformationen für diese Rolle zu erhalten. Anschließend konfiguriert er diese Anmeldeinformationen in Umgebungsvariablen, sodass nachfolgende AWS CLI Befehle mit den Berechtigungen der Rolle funktionieren. Solange David die Rolle verwendet, kann er nicht

seine Power-User-Privilegien im Entwicklungskonto verwenden, da immer nur eine Gruppe von Berechtigungen gleichzeitig gültig sein kann.

Beachten Sie, dass alle Zugriffsschlüssel und Token nur Beispiele sind und nicht wie hier dargestellt verwendet werden können. Ersetzen Sie sie mit den entsprechenden Werten aus Ihrer Live-Umgebung.

So übernehmen Sie eine Rolle

1. David öffnet ein Befehlszeilenfenster und bestätigt, dass der AWS CLI Client funktioniert, indem er den folgenden Befehl ausführt:

   ```
   aws help
   ```

   > **Note**
   >
   > Die Standardumgebung von David verwendet die Anmeldeinformationen des Benutzers `David` aus seinem Standardprofil, das er mit dem Befehl `aws configure` erstellt hat. Weitere Informationen finden Sie unter Konfigurieren der AWS Command Line Interface im AWS Command Line Interface -Leitfaden.

2. Er beginnt mit dem Verfahren zum Wechseln der Rolle, indem er folgenden Befehl ausführt, um zur Rolle `UpdateApp` in der Produktionsabrechnung zu wechseln. Der ARN der Rolle wurde ihm vom Administrator, der die Rolle erstellt hat, mitgeteilt. Der Befehl erfordert, dass Sie außerdem einen Sitzungsnamen angeben. Sie können hierfür einen beliebigen Text eingeben.

   ```
   aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateApp" --role-
   session-name "David-ProdUpdate"
   ```

   David erhält dann die folgende Ausgabe:

   ```
   {
       "Credentials": {
           "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
           "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
   +Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
   CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
   uZEXAMPLECihzFB5lTYLto9dyBgSDy
   ```

```
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

3. David sieht die drei Teile im Anmeldeinformationen-Abschnitt der Ausgabe, die er benötigt.

   - `AccessKeyId`
   - `SecretAccessKey`
   - `SessionToken`

   David muss die AWS CLI Umgebung so konfigurieren, dass diese Parameter bei nachfolgenden Aufrufen verwendet werden. Weitere Informationen zu den verschiedenen Möglichkeiten zum Konfigurieren Ihrer Anmeldeinformationen finden Sie unter Konfigurieren der AWS Command Line Interface. Den Befehl `aws configure` können Sie nicht verwenden, da er nicht die Erfassung des Sitzungs-Token unterstützt. Sie können jedoch die Informationen manuell in eine Konfigurationsdatei eingeben. Da die Ablaufzeit dieser temporären Anmeldeinformationen relativ kurz ist, fügen Sie sie besser der Umgebung Ihrer aktuellen Befehlszeilensitzung hinzu.

4. Um die drei Werte zur Umgebung hinzuzufügen, kopiert David die Ausgabe aus dem vorherigen Schritt und fügt sie in die folgenden Befehle ein. Fügen Sie die Ausgabe in einen einfachen Texteditor ein, um Probleme mit Zeilenumbrüchen in der Ausgabe des Sitzungs-Token zu vermeiden. Er muss als eine einzige lange Zeichenfolge hinzugefügt werden, auch wenn er hier aus Darstellungsgründen mit Zeilenumbrüchen angezeigt wird.

   > **Note**
   >
   > Das folgende Beispiel zeigt Befehle in der Windows-Umgebung, wobei „set“ der Befehl zum Erstellen einer Umgebungsvariablen ist. Auf einem Linux- oder macOS-Computer würden Sie stattdessen den Befehl für „Exportieren“ verwenden. Alle anderen Teile des Beispiels sind in allen drei Umgebungen gültig.
   > Details zur Verwendung von Tools für Windows PowerShell finden Sie hier: Zu einer IAM-Rolle wechseln (Tools für Windows PowerShell)

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
Ryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDyEXA
MPLEKEY9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLENhykxiHen
DHq6ikBQ==
```

Ab diesem Zeitpunkt werden alle folgenden Befehle mit den Berechtigungen der von diesen Anmeldeinformationen identifizierten Rolle ausgeführt. Im Fall von David handelt es sich um die Rolle `UpdateApp`.

5. Führen Sie den Befehl für den Zugriff auf die Ressourcen im Production-Konto. In diesem Beispiel listet David mit dem folgenden Befehl den Inhalt seines S3-Buckets auf.

```
aws s3 ls s3://productionapp
```

Da die Namen von Amazon-S3-Buckets universell eindeutig sind, muss die ID des Kontos, in dem der Bucket enthalten ist, nicht angegeben werden. Wenn Sie auf Ressourcen für andere AWS Dienste zugreifen möchten, finden Sie in der AWS CLI Dokumentation zu diesem Dienst die Befehle und die Syntax, die zum Verweisen auf seine Ressourcen erforderlich sind.

Verwenden von AssumeRole (AWS API)

Wenn David eine Aktualisierung des Codes für die Produktionsabrechnung vornehmen muss, führt er einen `AssumeRole`-Aufruf aus, um die `UpdateApp`-Rolle zu übernehmen. Der Aufruf gibt temporäre Anmeldeinformationen für den Zugriff auf den Bucket `productionapp` in der Produktionsabrechnung zurück. Mit diesen Anmeldeinformationen kann David API-Aufrufe ausführen, um den Bucket `productionapp` zu aktualisieren. Er kann jedoch keine API-Aufrufe für den Zugriff auf andere Ressourcen in der Produktionsabrechnung ausführen, selbst wenn er über Power-User-Privilegien im Entwicklungskonto verfügt.

So übernehmen Sie eine Rolle

1. David ruft `AssumeRole` als Teil einer Anwendung auf. Er muss den `UpdateApp`-ARN angeben: `arn:aws:iam::999999999999:role/UpdateApp`.

   Die Antwort auf den `AssumeRole`-Aufruf enthält die temporären Anmeldeinformationen mit einem `AccessKeyId` und einem `SecretAccessKey`. Sie enthält auch eine `Expiration`-Zeit, die angibt, wann die Anmeldeinformationen ablaufen und neue angefordert werden müssen.
2. Mit den temporären Anmeldeinformationen führt David einen Aufruf `s3:PutObject` aus, um den Bucket `productionapp` zu aktualisieren. Er übergibt die Anmeldeinformationen als `AuthParams`-Parameter an den API-Aufruf. Da die temporären Anmeldeinformationen der Rolle nur über Lese- und Schreibberechtigungen für den Bucket `productionapp` verfügen, werden alle andere Aktionen im Production-Konto verweigert.

Ein Code-Beispiel (mit Python) finden Sie unter Zu einer IAM-Rolle (AWS API) wechseln.

Zugehörige Ressourcen

- Weitere Informationen zu IAM-Benutzern und Gruppen finden Sie unter IAM-Identitäten (Benutzer, Gruppen und Rollen).
- Weitere Informationen über Amazon-S3-Buckets finden Sie unter Erstellen eines Buckets im Benutzerhandbuch für Amazon Simple Storage Service.
- Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

Übersicht

Sie haben das Tutorial für den kontoübergreifenden API-Zugriff abgeschlossen. Sie haben eine Rolle zum Einrichten einer Vertrauensstellung mit einem anderen Konto und zur Definition der Aktionen erstellt, zu deren Ausführung die vertrauenswürdigen Entitäten berechtigt sind. Anschließend haben Sie eine Gruppenrichtlinie geändert, um zu steuern, welche IAM-Benutzer auf die Rolle zugreifen können. Folglich können Entwickler aus dem Entwicklungskonto mithilfe von temporären Anmeldeinformationen Aktualisierungen am Bucket `productionapp` in der Produktionsabrechnung vornehmen.

IAM-Anleitung: Erstellen und Anfügen Ihrer ersten vom Kunden verwalteten Richtlinie

In diesem Tutorial verwenden Sie die, AWS Management Console um eine vom Kunden verwaltete Richtlinie zu erstellen und diese Richtlinie dann einem IAM-Benutzer in Ihrem anzuhängen. AWS-Konto Die von Ihnen erstellte Richtlinie ermöglicht es einem IAM-Testbenutzer, sich direkt AWS Management Console mit Leseberechtigungen bei der anzumelden.

Dieser Workflow umfasst drei grundlegende Schritte:

Schritt 1: Erstellen der Richtlinie

Standardmäßig haben IAM-Benutzer keine Berechtigungen zum Durchführen von Aktionen. Sie können nur auf die AWS -Managementkonsole zugreifen oder Daten darin verwalten, wenn Sie dies erlauben. In diesem Schritt erstellen Sie eine vom Kunden verwaltete Richtlinie, mit der sich angefügte Benutzer bei der Konsole anmelden können.

Schritt 2: Anfügen der Richtlinie

Wenn Sie eine Richtlinie an einen Benutzer anfügen, übernimmt dieser alle Zugriffsberechtigungen, die mit dieser Richtlinie verbunden sind. In diesem Schritt fügen Sie die neue Richtlinie an einen Testbenutzer an.

Schritt 3: Testen des Benutzerzugriffs

Nachdem die Richtlinie angefügt wurde, können Sie sich als Benutzer anmelden und die Richtlinie testen.

Voraussetzungen

Um die Schritte in dieser praktischen Anleitung auszuführen, müssen Sie bereits über Folgendes verfügen:

- Eine AWS-Konto , bei der Sie sich als IAM-Benutzer mit Administratorrechten anmelden können.
- Ein IAM-Testbenutzer, dem keine Berechtigungen zugewiesen wurden oder der über folgende Gruppenmitgliedschaften verfügt:

Benutzername	Gruppe	Berechtigungen
PolicyUser	<Keine>	<Keine>

Schritt 1: Erstellen der Richtlinie

In diesem Schritt erstellen Sie eine vom Kunden verwaltete Richtlinie, die es allen verbundenen Benutzern ermöglicht, sich AWS Management Console mit Lesezugriff auf IAM-Daten anzumelden.

So erstellen Sie die Richtlinie für Ihren Testbenutzer

1. Melden Sie sich bei der IAM-Konsole unter https://console.aws.amazon.com/iam/ mit Ihrem Benutzer an, der über Administratorrechte verfügt.
2. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie im Inhaltsbereich die Option Create policy (Richtlinie erstellen).
4. Wählen Sie die Option JSON aus und kopieren Sie den Text aus dem folgenden JSON-Richtliniendokument. Fügen Sie den folgenden Text in das JSON-Eingabefeld ein.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    } ]
}
```

5. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder auf der Registerkarte Visual editor die Option

> Review policy auswählen, strukturiert IAM möglicherweise Ihre Richtlinie neu, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

6. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) als Richtliniennamen **`UsersReadOnlyAccessToIAMConsole`** ein. Überprüfen Sie die von ihrer Richtlinie erteilten Berechtigungen und wählen Sie dann zum Speichern Ihrer Arbeit Create policy (Richtlinie erstellen) aus.

   Die neue Richtlinie wird in der Liste der verwalteten Richtlinien angezeigt und ist bereit.

Schritt 2: Anfügen der Richtlinie

Als Nächstes fügen Sie die soeben erstellte Richtlinie an Ihren IAM-Testbenutzer an.

So fügen Sie die Richtlinie an Ihren Testbenutzer an

1. Wählen Sie im Navigationsbereich der IAM-Konsole die Option Policies.
2. Beginnen Sie oben in der Richtlinienliste im Suchfeld mit der Eingabe von **`UsersReadOnlyAccesstoIAMConsole`**, bis Sie Ihre Richtlinie sehen. Wählen Sie dann in der Liste das Optionsfeld neben UsersReadOnlyAccessToIamConsole aus.
3. Wählen Sie die Schaltfläche Actions (Aktionen) und dann Attach (Anfügen).
4. Wählen Sie bei den IAM-Entitäten die Option zum Filtern nach Users (Benutzer).
5. Beginnen Sie im Suchfeld mit der Eingabe von **`PolicyUser`**, bis dieser Benutzer in der Liste angezeigt wird. Aktivieren Sie dann in der Liste das Kontrollkästchen neben diesem Benutzer.
6. Wählen Sie Richtlinie anfügen aus.

Sie haben die Richtlinie an Ihren IAM-Testbenutzer angefügt, was bedeutet, dass dieser Benutzer nun Lesezugriff auf die IAM-Konsole hat.

Schritt 3: Testen des Benutzerzugriffs

Für dieses Tutorial empfehlen wir Ihnen, den Zugriff zu testen, indem Sie sich als Testbenutzer anmelden, sodass Sie das potenzielle Erlebnis Ihrer Benutzer nachvollziehen können.

So testen Sie den Zugriff, indem Sie sich mit Ihrem Testbenutzer anmelden

1. Melden Sie sich mit Ihrem `PolicyUser`-Testbenutzer an der IAM-Konsole unter https://console.aws.amazon.com/iam/ an.
2. Navigieren Sie durch die Seiten der Konsole und versuchen Sie, einen neuen Benutzer oder eine neue Gruppe zu erstellen. Beachten Sie, dass der Testbenutzer `PolicyUser` zwar Daten anzeigen, aber keine IAM-Daten erstellen bzw. vorhandenen IAM-Daten ändern kann.

Zugehörige Ressourcen

Weitere Informationen finden Sie in den folgenden Ressourcen:

- Verwaltete Richtlinien und eingebundene Richtlinien
- Steuern des IAM-Benutzerzugriffs auf die AWS Management Console

Übersicht

Sie haben nun erfolgreich alle Schritte abgeschlossen, die zum Erstellen und Anfügen einer vom Kunden verwalteten Richtlinie erforderlich sind. Dadurch können Sie sich mit Ihrem Testkonto bei der IAM-Konsole anmelden, um zu sehen, wie die Erfahrung für Ihre Benutzer aussieht.

IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren

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-Ressourcen, einschließlich IAM-Entitäten (Benutzer oder Rollen), und an AWS Ressourcen anhängen. Sie können Richtlinien definieren, die Tag-Bedingungsschlüssel verwenden, um Ihren Auftraggeber basierend auf ihren Tags Berechtigungen zu erteilen. Wenn Sie Tags verwenden, um den Zugriff auf Ihre AWS Ressourcen zu kontrollieren, ermöglichen Sie es Ihren Teams und Ressourcen, mit weniger Änderungen an den Richtlinien zu AWS wachsen. ABAC-Richtlinien sind flexibler als herkömmliche AWS Richtlinien, bei denen Sie jede einzelne Ressource auflisten müssen. Weitere Informationen zu ABAC und seinen Vorteilen gegenüber herkömmlichen Richtlinien finden Sie unter Wofür ist ABAC? AWS.

> **Note**
>
> Sie müssen für jedes Sitzungs-Tag einen einzelnen Wert übergeben. AWS Security Token Service unterstützt keine mehrwertigen Sitzungs-Tags.

Themen

Tutorial-Übersicht

In diesem Tutorial wird gezeigt, wie Sie eine Richtlinie erstellen und testen, die es IAM-Rollen mit Auftraggeber-Tags ermöglicht, auf Ressourcen mit übereinstimmenden Tags zuzugreifen. Wenn ein Auftraggeber eine Anforderung an AWS stellt, werden seine Berechtigungen basierend auf der Tatsache erstellt, ob die Auftraggeber- und Ressourcen-Tags übereinstimmen. Diese Strategie ermöglicht es Einzelpersonen, nur die AWS Ressourcen anzuzeigen oder zu bearbeiten, die für ihre Arbeit erforderlich sind.

Szenario

Nehmen wir an, Sie sind leitender Entwickler bei einem großen Unternehmen mit dem Namen Example Corporation und erfahrener IAM-Administrator. Sie sind mit dem Erstellen und Verwalten von IAM-Benutzern, -Rollen und -Richtlinien vertraut. Sie möchten sicherstellen, dass Ihre

Entwicklungsingenieure und die Mitarbeiter des Qualitätssicherungsteams auf die benötigten Ressourcen zugreifen können. Sie benötigen außerdem eine Strategie, die sich an das Wachstum Ihres Unternehmens anpassen lässt.

Sie entscheiden sich dafür, AWS Ressourcen-Tags und IAM-Rollenprinzipal-Tags zu verwenden, um eine ABAC-Strategie für Dienste zu implementieren, die diese unterstützen, beginnend mit. AWS Secrets Manager Informationen dazu, welche Services eine Autorisierung basierend auf Tags unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Informationen darüber, welche Tagging-Bedingungsschlüssel Sie in einer Richtlinie mit den Aktionen und Ressourcen der einzelnen Dienste verwenden können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Dienste. AWS Sie können Ihren SAML-basierten Identitätsanbieter oder Webidentitätsanbieter so konfigurieren, dass Sitzungs-Tags an AWSübergeben werden. Wenn sich Ihre Mitarbeiter zusammenschließen AWS, werden ihre Attribute auf den daraus resultierenden Principal in angewendet. AWS Sie können dann ABAC verwenden, um Berechtigungen basierend auf diesen Attributen zuzulassen oder abzulehnen. Weitere Informationen dazu, wie die Verwendung von Sitzungs-Tags mit einer SAML-Verbundidentität von diesem Lernprogramm abweicht, finden Sie unter IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC.

Ihre Mitarbeiter im Bereich Engineering und Qualitätssicherung arbeiten entweder am Pegasus- oder am Unicorn-Projekt. Sie wählen die folgenden 3-stelligen Projekt- und Team-Tag-Werte aus:

- `access-project` = `peg` für das Pegasus-Projekt
- `access-project` = `uni` für das Unicorn-Projekt
- `access-team` = `eng` für das Engineering-Team
- `access-team` = `qas` für das Qualitätssicherungsteam

Darüber hinaus legen Sie fest, dass das Tag „`cost-center`Kostenzuweisung“ erforderlich sein soll, um benutzerdefinierte AWS Abrechnungsberichte zu aktivieren. Weitere Informationen finden Sie unter Verwendung von Kostenzuordnungs-Tags im AWS Billing and Cost Management Leitfaden.

Zusammenfassung der wichtigsten Entscheidungen

- Mitarbeiter melden sich mit IAM-Benutzeranmeldeinformationen an und übernehmen dann die IAM-Rolle für ihr Team und ihr Projekt. Wenn Ihr Unternehmen über ein eigenes Identitätssystem verfügt, können Sie einen Verbund einrichten, damit Mitarbeiter eine Rolle ohne IAM-Benutzer übernehmen können. Weitere Informationen finden Sie unter IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC.

- Es wird an allen Rollen dieselbe Richtlinie angefügt. Aktionen werden basierend auf Tags zugelassen oder verweigert.
- Mitarbeiter können neue Ressourcen erstellen, aber nur, wenn sie der Ressource dieselben Tags zuweisen, die auf ihre Rolle angewendet werden. Auf diese Weise wird sichergestellt, dass Mitarbeiter die Ressource anzeigen können, nachdem sie sie erstellt haben. Administratoren müssen Richtlinien nicht mehr mit dem ARN neuer Ressourcen aktualisieren.
- Mitarbeiter können unabhängig vom Projekt die Ressourcen lesen, die ihrem Team gehören.
- Mitarbeiter können Ressourcen aktualisieren und löschen, die zu ihrem Team und Projekt gehören.
- IAM-Administratoren können eine neue Rolle für neue Projekte hinzufügen. Sie können einen neuen IAM-Benutzer erstellen und markieren, um den Zugriff auf die entsprechende Rolle zu ermöglichen. Administratoren müssen keine Richtlinie bearbeiten, um ein neues Projekt oder Teammitglied zu unterstützen.

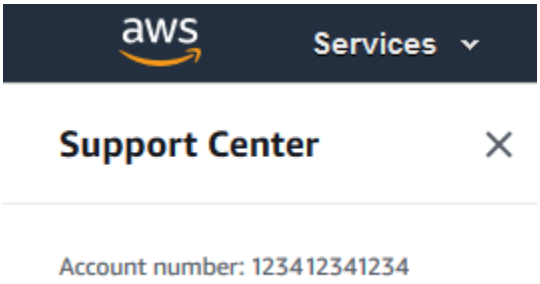
In diesem Tutorial markieren Sie alle Ressourcen sowie Ihre Projektrollen und fügen Richtlinien zu den Rollen hinzu, um das zuvor beschriebene Verhalten zuzulassen. Die resultierende Richtlinie erlaubt den Rollen `Create`, `Read`, `Update` und `Delete` den Zugriff auf Ressourcen, die mit denselben Projekt- und Team-Tags markiert sind. Die Richtlinie erlaubt auch einen projektübergreifenden `Read`-Zugriff auf Ressourcen, die mit demselben Team markiert sind.

Voraussetzungen

Um die Schritte in dieser praktischen Anleitung auszuführen, müssen Sie bereits über Folgendes verfügen:

- Und AWS-Konto bei dem Sie sich als Benutzer mit Administratorrechten anmelden können.
- Ihre 12-stellige Konto-ID, mit der Sie die Rollen in Schritt 3 erstellen.

  Um Ihre AWS Konto-ID-Nummer mithilfe von zu finden AWS Management Console, wählen Sie in der Navigationsleiste oben rechts Support und dann Support Center aus. Die Kontonummer (ID) wird im Navigationsbereich auf der linken Seite angezeigt.



- Erfahrung mit dem Erstellen und Bearbeiten von IAM-Benutzern, -Rollen und Richtlinien in der AWS Management Console. Wenn Sie jedoch Hilfe benötigen, um sich an einen IAM-Verwaltungsprozess zu erinnern, finden Sie in diesem Tutorial Links, über die Sie step-by-step Anweisungen einsehen können.

Schritt 1: Erstellen von Testbenutzern

Erstellen Sie zum Testen vier IAM-Benutzer mit Berechtigungen zum Übernehmen von Rollen mit denselben Tags. Dies erleichtert das Hinzufügen weiterer Benutzer zu Ihren Teams. Wenn Sie die Benutzer markieren, erhalten diese automatisch Zugriff, um die richtige Rolle übernehmen zu können. Sie müssen die Benutzer nicht zur Vertrauensrichtlinie der Rolle hinzufügen, wenn sie nur an einem Projekt und nur in einem Team arbeiten.

1. Erstellen Sie die folgende vom Kunden verwaltete Richtlinie namens `access-assume-role`. Weitere Informationen zum Erstellen einer JSON-Richtlinie finden Sie unter Erstellen von IAM-Richtlinien.

   ABAC-Richtlinie: Übernehmen einer beliebigen ABAC-Rolle, aber nur, wenn die Benutzer- und Rollen-Tags übereinstimmen

   Mit der folgenden Richtlinie kann ein Benutzer eine beliebige Rolle in Ihrem Konto mit dem `access-`-Namenspräfix übernehmen. Die Rolle muss mit denselben Projekt-, Team- und Kostenstellen-Tags wie der Benutzer markiert sein.

   Um diese Richtlinie zu verwenden, ersetzen Sie den kursiv gedruckten Platzhaltertext durch Ihre eigenen Kontoinformationen.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
   ```

```
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::account-ID-without-hyphens:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/access-
team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

Damit sich dieses Tutorial für eine große Anzahl von Benutzern eignet, können Sie die Richtlinie einer Gruppe anfügen und alle Benutzer zur Gruppe hinzufügen. Weitere Informationen finden Sie unter IAM-Benutzergruppen erstellen und Hinzufügen und Entfernen von Benutzern in einer IAM-Gruppe.

2. Erstellen Sie die folgenden IAM-Benutzer und fügen Sie die `access-assume-role`-Berechtigungsrichtlinie an. Stellen Sie sicher, dass Sie Benutzerzugriff auf AWS Management Console bereitstellen auswählen, und fügen Sie dann die folgenden Tags hinzu. Weitere Informationen zum Erstellen und Markieren eines neuen Benutzers finden Sie unter Erstellen von IAM-Benutzern (Konsole).

   ABAC-Benutzer

Benutzername	Benutzer-Tag-Schlüssel	Benutzer-Tag-Wert
`access-Arnav-peg-eng`	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`
`access-Mary-peg-qas`	`access-project`	`peg`

Benutzername	Benutzer-Tag-Schlüssel	Benutzer-Tag-Wert
	access-team cost-center	qas 987654
access-Saanvi-uni-eng	access-project access-team cost-center	uni eng 123456
access-Carlos-uni-qas	access-project access-team cost-center	uni qas 123456

Schritt 2: Erstellen der ABAC-Richtlinie

Erstellen Sie die folgende Richtlinie namens **`access-same-project-team`**. Sie fügen diese Richtlinie den Rollen in einem späteren Schritt hinzu. Weitere Informationen zum Erstellen einer JSON-Richtlinie finden Sie unter Erstellen von IAM-Richtlinien.

Weitere Richtlinien, die Sie für dieses Tutorial anpassen können, finden Sie auf den folgenden Seiten:

- Zugriffssteuerung für IAM-Auftraggeber
- Amazon EC2: Ermöglicht es, die von einem Benutzer markierten EC2-Instances programmgesteuert und in der Konsole zu starten oder zu stoppen
- EC2: Starten oder Stoppen von Instances basierend auf übereinstimmenden Auftraggeber- und Ressourcen-Tags
- EC2: Starten oder Stoppen von Instances basierend auf Tags
- IAM: Übernehmen von Rollen, die ein bestimmtes Tag haben

ABAC-Richtlinie: Zugriff nur auf Access Secrets Manager-Ressourcen, wenn die Auftraggeber- und Ressourcen-Tags übereinstimmen

Die folgende Richtlinie erlaubt es Auftraggeber, Ressourcen zu erstellen, zu lesen, zu bearbeiten und zu löschen, aber nur, wenn diese Ressourcen mit denselben Schlüssel-Wert-Paaren wie der Auftraggeber markiert sind. Wenn ein Auftraggeber eine Ressource erstellt, muss er die Tags `access-project`, `access-team` und `cost-center` mit Werten hinzufügen, die mit den Tags des Auftraggebers übereinstimmen. Die Richtlinie erlaubt auch das Hinzufügen optionaler `Name`- oder `0wnedBy`-Tags.

```
{
 "Version": "2012-10-17",
 "Statement": [
     {
         "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
         "Effect": "Allow",
         "Action": "secretsmanager:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
                 "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
             },
             "ForAllValues:StringEquals": {
                 "aws:TagKeys": [
                     "access-project",
                     "access-team",
                     "cost-center",
                     "Name",
                     "OwnedBy"
                 ]
             },
             "StringEqualsIfExists": {
                 "aws:RequestTag/access-project": "${aws:PrincipalTag/access-project}",
                 "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
             }
         }
     },
     {
         "Sid": "AllResourcesSecretsManagerNoTags",
         "Effect": "Allow",
         "Action": [
             "secretsmanager:GetRandomPassword",
```

```
            "secretsmanager:ListSecrets"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ReadSecretsManagerSameTeam",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:Describe*",
            "secretsmanager:Get*",
            "secretsmanager:List*"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
            }
        }
    },
    {
        "Sid": "DenyUntagSecretsManagerReservedTags",
        "Effect": "Deny",
        "Action": "secretsmanager:UntagResource",
        "Resource": "*",
        "Condition": {
            "ForAnyValue:StringLike": {
                "aws:TagKeys": "access-*"
            }
        }
    },
    {
        "Sid": "DenyPermissionsManagement",
        "Effect": "Deny",
        "Action": "secretsmanager:*Policy",
        "Resource": "*"
    }
 ]
}
```

Was macht diese Richtlinie?

- Die `AllActionsSecretsManagerSameProjectSameTeam`-Anweisung erlaubt alle Aktionen dieses Services für alle verwandten Ressourcen, aber nur, wenn die Ressourcen-Tags mit den

Auftraggeber-Tags übereinstimmen. Durch Hinzufügen von `"Action": "secretsmanager:*"` zur Richtlinie wächst die Richtlinie, wenn Secrets Manager wächst. Wenn Secrets Manager eine neue API-Operation hinzufügt, müssen Sie diese Aktion nicht zur Anweisung hinzufügen. Die Anweisung implementiert ABAC mithilfe von drei Bedingungsblöcken. Die Anforderung ist nur zulässig, wenn alle drei Blöcke "true" zurückgeben.

- Der erste Bedingungsblock dieser Anweisung gibt "true" zurück, wenn die angegebenen Tag-Schlüssel in der Ressource vorhanden sind und ihre Werte mit den Tags des Auftraggebers übereinstimmen. Dieser Block gibt „false“ zurück, wenn die Tags nicht übereinstimmen oder wenn Aktionen das Ressourcen-Tagging nicht unterstützen. Informationen darüber, welche Aktionen in diesem Block nicht zulässig sind, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Secrets Manager. Diese Seite zeigt, dass Aktionen, die für den Secret-Ressourcentyp durchgeführt werden, den `secretsmanager:ResourceTag/tag-key`-Bedingungsschlüssel unterstützen. Einige Secrets Manager-Aktionen unterstützen diesen Ressourcentyp nicht, einschließlich `GetRandomPassword` und `ListSecrets`. Sie müssen zusätzliche Anweisungen erstellen, um diese Aktionen zuzulassen.
- Der zweite Bedingungsblock gibt "true" zurück, wenn jeder Tag-Schlüssel, der in der Anforderung übergeben wurde, in der angegebenen Liste enthalten ist. Dies erfolgt unter Verwendung von `ForAllValues` mit dem `StringEquals`-Bedingungsoperator. Wenn keine Schlüssel oder keine Teilmenge des Schlüsselsatzes übergeben werden, gibt die Bedingung „true“ zurück. Dies ermöglicht `Get*`-Operationen, die keine Übergabe von Tags in der Anforderung zulassen. Wenn der Anforderer einen Tag-Schlüssel enthält, der nicht in der Liste enthalten ist, gibt die Bedingung "false" zurück. Jeder Tag-Schlüssel, der in der Anforderung übergeben wird, muss mit einem Mitglied dieser Liste übereinstimmen. Weitere Informationen finden Sie unter Mehrwertige Kontextschlüssel.
- Der dritte Bedingungsblock gibt „true“ zurück, wenn die Anforderung die Übergabe von Tags unterstützt, wenn alle drei Tags vorhanden sind und wenn sie mit den Auftraggeber-Tag-Werten übereinstimmen. Dieser Block gibt auch "true" zurück, wenn die Anforderung die Übergabe von Tags nicht unterstützt. Dies liegt an `...IfExists` im Bedingungsoperator. Der Block gibt "false" zurück, wenn während einer Aktion, die das unterstützt, kein Tag übergeben wird oder wenn die Tag-Schlüssel und -Werte nicht übereinstimmen.

- Die `AllResourcesSecretsManagerNoTags`-Anweisung erlaubt die Aktionen `ListSecrets` und `GetRandomPassword`, die von der ersten Anweisung nicht zugelassen werden.
- Die `ReadSecretsManagerSameTeam`-Anweisung erlaubt schreibgeschützte Operationen, wenn der Auftraggeber mit demselben access-team-Tag wie die Ressource markiert ist. Dies ist unabhängig vom Projekt- oder Kostenstellen-Tag zulässig.

- Die `DenyUntagSecretsManagerReservedTags`-Anweisung lehnt Anforderungen zum Entfernen von Tags mit Schlüsseln, die mit "access-" beginnen, aus Secrets Manager ab. Diese Tags werden verwendet, um den Zugriff auf Ressourcen zu steuern. Das Entfernen von Tags kann daher dazu führen, dass auch Berechtigungen entfernt werden.
- Die `DenyPermissionsManagement`-Anweisung verweigert den Zugriff zum Erstellen, Bearbeiten oder Löschen von ressourcenbasierten Secrets Manager-Richtlinien. Diese Richtlinien können verwendet werden, um die Berechtigungen des Secrets zu ändern.

> **Important**
>
> Diese Richtlinie verwendet eine Strategie, bei der alle Aktionen für einen Service zugelassen werden, es sei denn, es handelt sich um Aktionen, durch die Berechtigungen geändert werden. Wird eine Aktion verweigert, wird jede andere Richtlinie außer Kraft gesetzt, die es dem Auftraggeber erlaubt, diese Aktion auszuführen. Dies kann unerwünschte Ergebnisse nach sich ziehen. Es hat sich bewährt, explizite Zugriffsverweigerungen nur einzusetzen, wenn es keine Umstände gibt, die dazu führen, dass diese Aktion zugelassen werden sollte. Lassen Sie andernfalls eine Liste einzelner Aktionen zu, und die unerwünschten Aktionen werden standardmäßig verweigert.

Schritt 3: Erstellen von Rollen

Erstellen Sie die folgenden IAM-Rollen und fügen Sie die **`access-same-project-team`**-Richtlinie an, die Sie im vorherigen Schritt erstellt haben. Weitere Informationen zum Erstellen einer IAM-Rolle finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer. Informationen zur Verwendung eines Verbunds anstelle von IAM-Benutzern und -Rollen finden Sie unter IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC.

ABAC-Rollen

Auftragsfunktion	Rollenname	Rollen-Tags	Rollenbeschreibung
Projekt Pegasus Engineering	`access-peg-engineering`	`access-project = peg`	Ermöglicht es Ingenieuren, alle Engineering-Ressourcen zu lesen und Pegasus-

Auftragsfunktion	Rollenname	Rollen-Tags	Rollenbeschreibung
		`access-team = eng` `cost-center = 987654`	Engineering-Ressourcen zu erstellen und zu verwalten.

Qualitätssicherung für das Pegasus-Projekt	`access-peg-quality-assurance`	`access-project = peg`
		`access-team = qas`
		`cost-center = 987654`
			Ermöglicht dem QA-Team, alle QA-Ressourcen zu lesen und alle QA-Ressourcen des Pegasus-Projekts zu erstellen und zu verwalten.

Projekt Unicorn Engineering	`access-uni-engineering`	`access-project = uni`
		`access-team = eng`
		`cost-center = 123456`
			Ermöglicht es Ingenieuren, alle Engineering-Ressourcen zu lesen und alle Engineering-Ressourcen des Unicorn-Projekts zu erstellen und zu verwalten.

Auftragsfunktion	Rollenname	Rollen-Tags	Rollenbeschreibung

Qualitätssicherung für Project Unicorn	`access-uni-quality-assurance`	`access-project = uni`
		
		`access-team = qas`
		
		`cost-center = 123456`
			Ermöglicht dem QA-Team, alle QA-Ressourcen zu lesen und alle QA-Ressourcen des Unicorn-Projekts zu erstellen und zu verwalten.

Schritt 4: Testen der Erstellung von Secrets

Die Berechtigungsrichtlinie, die den Rollen angefügt ist, ermöglicht es den Mitarbeitern, Secrets zu erstellen. Dies ist nur zulässig, wenn das Secret mit dem Projekt, dem Team und der Kostenstelle markiert ist. Vergewissern Sie sich, dass Ihre Berechtigungen wie erwartet funktionieren, indem Sie sich als Ihre Benutzer anmelden, die richtige Rolle übernehmen und Aktivitäten in Secrets Manager testen.

So testen Sie das Erstellen eines Secrets mit und ohne die erforderlichen Tags

1. Bleiben Sie im Hauptbrowserfenster als Administratorbenutzer angemeldet, sodass Sie Benutzer, Rollen und Richtlinien in IAM überprüfen können. Verwenden Sie ein Browser-Inkognito-Fenster oder einen separaten Browser für Ihre Tests. Melden Sie sich dort als `access-Arnav-peg-eng`-IAM-Benutzer und öffnen Sie die Secrets Manager Konsole unterhttps://console.aws.amazon.com/secretsmanager/.
2. Versuchen Sie, zur `access-uni-engineering`-Rolle zu wechseln.

   Diese Operation schlägt fehl, da die `access-project`- und `cost-center`-Tag-Werte für den Benutzer `access-Arnav-peg-eng` und die Rolle `access-uni-engineering` nicht übereinstimmen.

   Weitere Informationen zum Rollenwechsel finden Sie in AWS Management ConsoleWechseln zu einer Rolle (Konsole)

3. Wechseln Sie zur `access-peg-engineering`-Rolle.
4. Speichern Sie ein neues Secret mit den folgenden Informationen. Weitere Informationen zum Speichern eines Secrets finden Sie unter Erstellen eines Basis-Secrets im AWS Secrets Manager -Leitfaden.

> **Important**
>
> Secrets Manager zeigt Warnmeldungen an, da Sie keine Berechtigungen für zusätzliche AWS -Services haben, die mit Secrets Manager funktionieren. Um beispielsweise Anmeldeinformationen für eine Amazon RDS-Datenbank zu erstellen, benötigen Sie die Berechtigung zum Beschreiben von RDS-Instances, sowie RDS- und Amazon Redshift-Clustern. Sie können diese Benachrichtigungen ignorieren, da Sie diese speziellen AWS Dienste in diesem Tutorial nicht verwenden.

1. Wählen Sie im Abschnitt Select secret type (Secret-Typ auswählen) die Option Other type of secrets (Andere Secret-Typen). Geben Sie in die beiden Textfelder `test-access-key` und `test-access-secret` ein.
2. Geben Sie `test-access-peg-eng` in das Feld Secret name (Secret-Name) ein.
3. Fügen Sie verschiedene Tag-Kombinationen aus der folgenden Tabelle hinzu und überprüfen Sie das erwartete Verhalten.
4. Wählen Sie Store (Speichern) aus, um zu versuchen, das Secret zu erstellen. Kehren Sie zu den vorherigen Seiten der Secrets Manager-Konsole zurück, wenn das Speichern fehlschlägt, und verwenden Sie den nächsten Tag-Satz aus der folgenden Tabelle. Der letzte Tag-Satz ist zulässig und erstellt erfolgreich das Secret.

ABAC-Tag-Kombinationen für die **`test-access-peg-eng`**-Rolle

`access-project` - Tag-Wert	**`access-team`** - Tag-Wert	**`cost-center`** - Tag-Wert	Zusätzliche Tags	Erwartetes Verhalten
(Keine)	(Keine)	(Keine)	(Keine)	Verweigert, da der `access-project` -Tag-Wert nicht mit dem Wert der Rolle von `peg` übereinstimmt.

access-project - Tag-Wert	**access-team** - Tag-Wert	**cost-center** - Tag-Wert	Zusätzliche Tags	Erwartetes Verhalten
uni	eng	987654	(Keine)	Verweigert, da der access-project -Tag-Wert nicht mit dem Wert der Rolle von peg übereinstimmt.
peg	qas	987654	(Keine)	Verweigert, da der access-team -Tag-Wert nicht mit dem Wert der Rolle von eng übereinstimmt.
peg	eng	123456	(Keine)	Verweigert, da der cost-center -Tag-Wert nicht mit dem Wert der Rolle von 987654 übereinstimmt.
peg	eng	987654	owner = Jane	Verweigert, da die Richtlinie das zusätzliche Tag owner nicht zulässt, auch wenn alle drei erforderlichen Tags vorhanden sind und ihre Werte mit den Werten der Rolle übereinstimmen.
peg	eng	987654	Name = Jane	Zulässig, da alle drei erforderlichen Tags vorhanden sind und ihre Werte mit den Werten der Rolle übereinstimmen. Sie können auch das optionale Name-Tag mit einschließen.

5. Melden Sie sich ab und wiederholen Sie die ersten drei Schritte dieses Verfahrens für alle folgenden Rollen und Tag-Werte. Testen Sie im vierten Schritt dieses Verfahrens alle von Ihnen ausgewählten fehlenden Tags, optionalen Tags, unzulässigen Tags und ungültigen Tag-Werte. Verwenden Sie dann die erforderlichen Tags, um ein Secret mit den folgenden Tags und dem folgenden Namen zu erstellen.

ABAC-Rollen und -Tags

Benutzername	Rollenname	Secret-Name	Secret-Tags
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654
access-Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = uni access-team = qas cost-center = 123456

Schritt 5: Testen der Anzeige von Secrets

Die Richtlinie, die Sie an die einzelnen Rollen angefügt haben, ermöglicht den Mitarbeitern unabhängig vom Projekt, alle Secrets anzuzeigen, die mit ihrem Teamnamen markiert sind. Vergewissern Sie sich, dass Ihre Berechtigungen wie erwartet funktionieren, indem Sie Ihre Rollen in Secrets Manager testen.

So testen Sie die Anzeige eines Secrets mit und ohne die erforderlichen Tags

1. Melden Sie sich als einer der folgenden IAM-Benutzer an:

   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`

2. Wechseln Sie zur übereinstimmenden Rolle:

   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-uni-quality-assurance`

   Weitere Informationen zum Rollenwechsel in der AWS Management Console finden Sie unterWechseln zu einer Rolle (Konsole).

3. Klicken Sie im Navigationsbereich auf der linken Seite auf das Menüsymbol, um das Menü zu erweitern, und wählen Sie dann Secrets aus.

4. Sie sollten alle vier Secrets in der Tabelle sehen, unabhängig von Ihrer aktuellen Rolle. Das wird erwartet, da die Richtlinie namens `access-same-project-team` die Aktion `secretsmanager:ListSecrets` für alle Ressourcen zulässt.

5. Wählen Sie den Namen eines der Secrets.

6. Auf der Detailseite für das Secret bestimmen die Tags Ihrer Rolle, ob Sie den Seiteninhalt anzeigen können. Vergleichen Sie den Namen Ihrer Rolle mit dem Namen Ihres Secrets. Wenn sie denselben Teamnamen haben, stimmen die `access-team`-Tags überein. Wenn sie nicht übereinstimmen, wird der Zugriff verweigert.

   ABAC-Secret-Anzeigeverhalten für die einzelnen Rollen

Rollenname	Secret-Name	Erwartetes Verhalten
`access-peg-engineering`	`test-access-peg-eng`	Zulässig
	`test-access-peg-qas`	Nicht zulässig

Rollenname	Secret-Name	Erwartetes Verhalten
	`test-access-uni-eng`	Zulässig
	`test-access-uni-qas`	Nicht zulässig
`access-peg-quality-assurance`	`test-access-peg-eng`	Nicht zulässig
	`test-access-peg-qas`	Zulässig
	`test-access-uni-eng`	Nicht zulässig
	`test-access-uni-qas`	Zulässig
`access-uni-engineering`	`test-access-peg-eng`	Zulässig
	`test-access-peg-qas`	Nicht zulässig
	`test-access-uni-eng`	Zulässig
	`test-access-uni-qas`	Nicht zulässig
`access-uni-quality-assurance`	`test-access-peg-eng`	Nicht zulässig
	`test-access-peg-qas`	Zulässig
	`test-access-uni-eng`	Nicht zulässig
	`test-access-uni-qas`	Zulässig

7. Wählen Sie aus den Breadcrumbs oben auf der Seite Secrets aus, um zur Liste der Secrets zurückzukehren. Wiederholen Sie die Schritte in diesem Verfahren mit verschiedenen Rollen, um zu testen, ob Sie alle Secrets anzeigen können.

Schritt 6: Testen der Skalierbarkeit

Ein wichtiger Grund für die Verwendung der attributbasierten Zugriffskontrolle (ABAC) anstelle der rollenbasierten Zugriffskontrolle (RBAC) ist die Skalierbarkeit. Wenn Ihr Unternehmen neue Projekte, Teams oder Mitarbeiter hinzufügt AWS, müssen Sie Ihre ABAC-basierten Richtlinien nicht aktualisieren. Nehmen wir beispielsweise an, dass Example Company ein neues Projekt mit dem Namen Centaur finanziert. Eine Ingenieurin namens Saanvi Sarkar wird die leitende Ingenieurin

für das Centaur-Projekt sein, aber weiterhin am Unicorn-Projekt mitarbeiten. Saanvi wird auch die Arbeiten für das Peg-Projekt überprüfen. Es gibt auch mehrere neu eingestellte Ingenieure, darunter Nikhil Jayashankar, die nur am Centaur-Projekt arbeiten werden.

Um das neue Projekt hinzuzufügen AWS

1. Melden Sie sich als IAM-Administratorbenutzer an und öffnen Sie die -Konsole unterhttps://console.aws.amazon.com/iam/.
2. Wählen Sie links im Navigationsbereich Roles (Rollen) aus und fügen Sie eine IAM-Rolle namens `access-cen-engineering` hinzu. Fügen Sie die **`access-same-project-team`**-Berechtigungsrichtlinie an die Rolle an und fügen Sie folgenden Rollen-Tags hinzu:

   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
3. Wählen Sie im Navigationsbereich auf der linken Seite Users (Benutzer).
4. Fügen Sie einen neuen Benutzer mit dem Namen `access-Nikhil-cen-eng` hinzu, fügen Sie die Richtlinie namens `access-assume-role` an und fügen Sie die folgenden Benutzer-Tags hinzu.

   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
5. Verwenden Sie die Verfahren in Schritt 4: Testen der Erstellung von Secrets und Schritt 5: Testen der Anzeige von Secrets. Vergewissern Sie sich in einem anderen Browserfenster, dass Nikhil nur Centaur-Engineering-Secrets erstellen kann und dass er alle Engineering-Secrets anzeigen kann.
6. Wählen Sie im Hauptfenster des Browsers, in dem Sie sich als Administrator angemeldet haben, den Benutzer `access-Saanvi-uni-eng`.
7. Entfernen Sie auf der Registerkarte access-assume-roleBerechtigungen die Berechtigungsrichtlinie.
8. Fügen Sie die folgende Inlinerichtlinie namens `access-assume-specific-roles` hinzu. Weitere Informationen zum Hinzufügen einer Inlinerichtlinie zu einem Benutzer finden Sie unter So betten Sie eine eingebundene Richtlinie für einen Benutzer oder eine Rolle ein (Konsole).

   ABAC-Richtlinie: Nur bestimmte Rollen übernehmen

Diese Richtlinie erlaubt Saanvi, die Engineering-Rollen für das Pegasus- oder Centaur-Projekt anzunehmen. Diese benutzerdefinierte Richtlinie muss erstellt werden, da IAM keine mehrwertigen Tags unterstützt. Sie können den Benutzernamen von Saanvi nicht mit `access-project` = `peg` und `access-project` = `cen` markieren. Darüber hinaus kann das AWS Autorisierungsmodell nicht beiden Werten entsprechen. Weitere Informationen finden Sie unter Regeln für das Tagging in IAM und AWS STS. Stattdessen müssen Sie die beiden Rollen, die sie übernehmen kann, manuell angeben.

Um diese Richtlinie zu verwenden, ersetzen Sie den kursiv gedruckten Platzhaltertext durch Ihre eigenen Kontoinformationen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::account-ID-without-hyphens:role/access-peg-
engineering",
                "arn:aws:iam::account-ID-without-hyphens:role/access-cen-
engineering"
            ]
        }
    ]
}
```

9. Verwenden Sie die Verfahren in Schritt 4: Testen der Erstellung von Secrets und Schritt 5: Testen der Anzeige von Secrets. Bestätigen Sie in einem anderen Browserfenster, dass Saanvi beide Rollen übernehmen kann. Stellen Sie sicher, dass sie je nach den Tags der Rolle nur für ihr Projekt, ihr Team und ihre Kostenstelle Secrets erstellen kann. Bestätigen Sie auch, dass sie Details zu allen Secrets des Engineering-Teams anzeigen kann, einschließlich der Secrets, die sie gerade erstellt hat.

Schritt 7: Testen des Aktualisierens und Löschens von Secrets

Die `access-same-project-team`-Richtlinie, die den Rollen angefügt ist, ermöglicht es den Mitarbeitern, alle Secrets zu aktualisieren und zu löschen, die mit ihrem Projekt, ihrem Team und

ihrer Kostenstelle markiert sind. Vergewissern Sie sich, dass Ihre Berechtigungen wie erwartet funktionieren, indem Sie Ihre Rollen in Secrets Manager testen.

So testen Sie das Aktualisieren und Löschen eines Secrets mit und ohne die erforderlichen Tags

1. Melden Sie sich als einer der folgenden IAM-Benutzer an:

   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
   - `access-Nikhil-cen-eng`

2. Wechseln Sie zur übereinstimmenden Rolle:

   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-peg-quality-assurance`
   - `access-cen-engineering`

   Weitere Informationen zum Rollenwechsel finden Sie AWS Management Console unterWechseln zu einer Rolle (Konsole).

3. Versuchen Sie für jede Rolle, die Secret-Beschreibung zu aktualisieren, und versuchen Sie dann, die folgenden Secrets zu löschen. Weitere Informationen finden Sie unter Ändern eines Secrets und Löschen und Wiederherstellen eines Secrets im AWS Secrets Manager -Leitfaden.

   Aktualisierungs- und Löschverhalten eines ABAC-Secrets für die einzelnen Rollen

Rollenname	Secret-Name	Erwartetes Verhalten
`access-peg-engineering`	`test-access-peg-eng`	Zulässig
	`test-access-uni-eng`	Nicht zulässig
	`test-access-uni-qas`	Nicht zulässig

Rollenname	Secret-Name	Erwartetes Verhalten
`access-peg-quality-assurance`	`test-access-peg-qas`	Zulässig
	`test-access-uni-eng`	Nicht zulässig
`access-uni-engineering`	`test-access-uni-eng`	Zulässig
	`test-access-uni-qas`	Nicht zulässig
`access-peg-quality-assurance`	`test-access-uni-qas`	Zulässig

Übersicht

Sie haben nun alle erforderlichen Schritte zur Verwendung von Tags für die attributbasierte Zugriffskontrolle (ABAC) erfolgreich abgeschlossen. Sie haben gelernt, wie Sie eine Tagging-Strategie definieren. Sie haben diese Strategie auf Ihre Auftraggeber und Ressourcen angewendet. Sie haben eine Richtlinie erstellt und angewendet, die die Strategie für Secrets Manager durchsetzt. Sie haben auch gelernt, dass ABAC einfach skaliert werden kann, wenn Sie neue Projekte und Teammitglieder hinzufügen möchten. Daher können Sie sich mit Ihren Testrollen bei der IAM-Konsole anmelden und erfahren, wie Sie Tags für ABAC in AWS verwenden.

> **Note**
>
> Sie haben Richtlinien hinzugefügt, die Aktionen nur unter bestimmten Bedingungen zulassen. Wenn Sie eine andere Richtlinie auf Ihre Benutzer oder Rollen anwenden, die über breitere Berechtigungen verfügen, sind die Aktionen möglicherweise nicht auf Tagging beschränkt. Wenn Sie einem Benutzer beispielsweise mithilfe der `AdministratorAccess` AWS verwalteten Richtlinie vollständige Administratorrechte gewähren, wird dieser Zugriff durch diese Richtlinien nicht eingeschränkt. Weitere Informationen dazu, wie Berechtigungen festgelegt werden, wenn mehrere Richtlinien beteiligt sind, finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird.

Zugehörige Ressourcen

Weitere Informationen finden Sie in den folgenden Ressourcen:

- Wofür ist ABAC? AWS
- AWS Kontextschlüssel für globale Bedingungen
- Erstellen von IAM-Benutzern (Konsole)
- Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer
- Markieren von IAM-Ressourcen
- Steuern des Zugriffs auf AWS Ressourcen mithilfe von Tags
- Wechseln zu einer Rolle (Konsole)
- IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC

Informationen zur Überwachung der Tags in Ihrem Konto finden Sie unter Überwachen von Tag-Änderungen auf AWS Ressourcen mit serverlosen Workflows und Amazon CloudWatch Events.

IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC

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-Ressourcen, einschließlich IAM-Entitäten (Benutzer oder Rollen), und an AWS Ressourcen anhängen. Wenn die Entitäten für Anfragen verwendet werden, werden sie zu Principals AWS, und diese Principals enthalten Tags.

Sie können Sitzungs-Tags auch übergeben, wenn Sie eine Rolle übernehmen oder einen Benutzer in einen Verbund aufnehmen. Anschließend können Sie Richtlinien definieren, die Tag-Bedingungsschlüssel verwenden, um Ihren Auftraggeber basierend auf ihren Tags Berechtigungen zu erteilen. Wenn Sie Tags verwenden, um den Zugriff auf Ihre AWS -Ressourcen zu steuern, erlauben Sie Ihren Teams und Ressourcen das Wachsen, ohne dass viele Änderungen an AWS -Richtlinien vorgenommen werden müssen. ABAC-Richtlinien sind flexibler als herkömmliche AWS Richtlinien, bei denen Sie jede einzelne Ressource auflisten müssen. Weitere Informationen zu ABAC und seinen Vorteilen gegenüber herkömmlichen Richtlinien finden Sie unter Wofür ist ABAC? AWS.

Wenn Ihr Unternehmen einen SAML-basierten Identitätsanbieter (IdP) verwendet, um Unternehmens-Benutzeridentitäten zu verwalten, können Sie SAML-Attribute für eine differenzierte Zugriffskontrolle in AWS verwenden. Zu den Attributen können Kostenstellenkennungen, Benutzer-E-Mail-Adressen, Abteilungsklassifizierungen und Projektzuweisungen gehören. Wenn Sie diese Attribute als Sitzungs-Tags übergeben, können Sie den Zugriff auf AWS basierend auf diesen Sitzungs-Tags steuern.

Um zum Abschluss des ABAC-Tutorials SAML-Attribute an den Sitzungsauftraggeber zu übergeben, führen Sie die Aufgaben in IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren mit den in diesem Thema enthaltenen Änderungen.

Voraussetzungen

Um die Schritte zur Verwendung von SAML-Sitzungs-Tags für ABAC durchzuführen, müssen Sie bereits über Folgendes verfügen:

- Zugriff auf einen SAML-basierten Identitätsanbieter, bei dem Sie Testbenutzer mit bestimmten Attributen erstellen können.
- Die Fähigkeit zum Anmelden als ein Benutzer mit Administratorberechtigung.
- Erfahrung mit dem Erstellen und Bearbeiten von IAM-Benutzern, -Rollen und Richtlinien in der AWS Management Console. Wenn Sie jedoch Hilfe benötigen, um sich an einen IAM-Verwaltungsprozess zu erinnern, finden Sie im ABAC-Tutorial Links, über die Sie Anweisungen einsehen können. step-by-step
- Erfahrung mit dem Einrichten eines SAML-basierten Identitätsanbieters in IAM. Weitere Details und Links zur ausführlichen IAM-Dokumentation finden Sie unter Übergabe von Sitzungs-Tags mit AssumeRoleWith SAML.

Schritt 1: Erstellen von Testbenutzern

Überspringen Sie die Anweisungen in Schritt 1: Erstellen von Testbenutzern. Da Ihre Identitäten bei Ihrem Anbieter definiert sind, müssen Sie keine IAM-Benutzer für Ihre Mitarbeiter hinzufügen.

Schritt 2: Erstellen der ABAC-Richtlinie

Befolgen Sie die Anweisungen unter Schritt 2: Erstellen der ABAC-Richtlinie, um die angegebene verwaltete Richtlinie in IAM zu erstellen.

Schritt 3: Erstellen und Konfigurieren der SAML-Rolle

Wenn Sie das ABAC-Tutorial für SAML verwenden, müssen Sie zusätzliche Schritte ausführen, um die Rolle zu erstellen, den SAML-IdP zu konfigurieren und den Zugriff zu aktivieren. AWS Management Console Weitere Informationen finden Sie unter Schritt 3: Erstellen von Rollen.

Schritt 3A: Erstellen der SAML-Rolle

Erstellen Sie eine einzelne Rolle, die Ihrem SAML-Identitätsanbieter und dem IAM-Benutzer `test-session-tags` vertraut, den Sie in Schritt 1 erstellt haben. Das ABAC-Tutorial verwendet separate Rollen mit unterschiedlichen Rollen-Tags. Da Sie Sitzungs-Tags von Ihrem SAML-Identitätsanbieter übergeben, benötigen Sie nur eine Rolle. Informationen zum Erstellen einer SAML-basierten Rolle finden Sie unter Eine Rolle für den SAML 2.0-Verbund erstellen (Konsole).

Benennen Sie die Rolle `access-session-tags`. Fügen Sie die Berechtigungsrichtlinie `access-same-project-team` der Rolle an. Bearbeiten Sie die Rollenvertrauensrichtlinie, um die folgende Richtlinie zu verwenden. Ausführliche Anweisungen zum Bearbeiten der Vertrauensstellung einer Rolle finden Sie unter Ändern einer Rolle (Konsole).

Mit der folgenden Rollenvertrauensrichtlinie können Ihr SAML-Identitätsanbieter und der `test-session-tags`-Benutzer die Rolle übernehmen. Wenn sie die Rolle übernehmen, müssen sie die drei angegebenen Sitzungs-Tags übergeben. Die `sts:TagSession`-Aktion ist erforderlich, um die Übergabe von Sitzungs-Tags zu ermöglichen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/saml"}
            }
        }
```

```
    ]
}
```

Die `AllowSamlIdentityAssumeRole` Erklärung ermöglicht es Mitgliedern der Engineering- und Qualitätssicherungsteams, diese Rolle zu übernehmen, wenn sie sich zum IdP AWS der Example Corporation zusammenschließen. Der SAML-Anbieter `ExampleCorpProvider` ist in IAM definiert. Der Administrator hat die SAML-Zusicherung bereits so eingerichtet, dass die drei erforderlichen Sitzungs-Tags übergeben werden. Die Zusicherung kann zusätzliche Tags übergeben, aber diese drei müssen vorhanden sein. Die Attribute der Identität können einen beliebigen Wert für die Tags`cost-center` und `access-project` haben. Der Wert des Attributs `access-team` muss jedoch `eng` oder `qas` entsprechen, um anzugeben, dass sich die Identität im Engineering- oder Qualitätssicherungsteam befindet.

Schritt 3B: Konfigurieren des SAML-Identitätsanbieters

Konfigurieren Sie Ihren SAML-Identitätsanbieter so, dass die Attribute `cost-center`, `access-project` und`access-team` als Sitzungs-Tags übergeben werden. Weitere Informationen finden Sie unter Übergabe von Sitzungs-Tags mit AssumeRoleWith SAML.

Um diese Attribute als Sitzungs-Tags zu übergeben, schließen Sie die folgenden Elemente in Ihre SAML-Zusicherung ein.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
  <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
  <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
  <AttributeValue>eng</AttributeValue>
</Attribute>
```

Schritt 3C: Aktivieren des Konsolenzugriffs

Aktivieren Sie den Konsolenzugriff für Ihre verbundenen SAML-Benutzer. Weitere Informationen finden Sie unter Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console.

Schritt 4: Testen der Erstellung von Secrets

Schließen Sie sich der Rolle an, die diese Rolle AWS Management Console verwendet. `access-session-tags` Weitere Informationen finden Sie unter Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console. Folgen Sie dann den Anweisungen in Schritt 4: Testen der Erstellung von Secrets, um Secrets zu erstellen. Verwenden Sie verschiedene SAML-Identitäten mit Attributen, um den im ABAC-Tutorial angegebenen Tags zu entsprechen. Weitere Informationen finden Sie unter Schritt 4: Testen der Erstellung von Secrets.

Schritt 5: Testen der Anzeige von Secrets

Folgen Sie den Anweisungen in Schritt 5: Testen der Anzeige von Secrets, um die Secrets anzuzeigen, die Sie im vorherigen Schritt erstellt haben. Verwenden Sie verschiedene SAML-Identitäten mit Attributen, um den im ABAC-Tutorial angegebenen Tags zu entsprechen.

Schritt 6: Testen der Skalierbarkeit

Befolgen Sie die Anweisungen in Schritt 6: Testen der Skalierbarkeit, um die Skalierbarkeit zu testen. Fügen Sie dazu eine neue Identität mit den folgenden Attributen bei Ihrem SAML-basierten Identitätsanbieter hinzu:

- `cost-center = 101010`
- `access-project = cen`
- `access-team = eng`

Schritt 7: Testen des Aktualisierens und Löschens von Secrets

Folgen Sie den Anweisungen in Schritt 7: Testen des Aktualisierens und Löschens von Secrets, um Secrets zu aktualisieren und zu löschen. Verwenden Sie verschiedene SAML-Identitäten mit Attributen, um den im ABAC-Tutorial angegebenen Tags zu entsprechen.

> **Important**
>
> Löschen Sie alle Secrets, die Sie erstellt haben, um Abrechnungsgebühren zu vermeiden. Informationen zu den Preisen in Secrets Manager finden Sie unter AWS Secrets Manager - Preise.

Übersicht

Sie haben nun alle Schritte erfolgreich abgeschlossen, die erforderlich sind, um SAML-Sitzungstags und Ressourcen-Tags für die Berechtigungsverwaltung verwenden zu können.

> **Note**
>
> Sie haben Richtlinien hinzugefügt, die Aktionen nur unter bestimmten Bedingungen zulassen. Wenn Sie eine andere Richtlinie auf Ihre Benutzer oder Rollen anwenden, die über breitere Berechtigungen verfügen, sind die Aktionen möglicherweise nicht auf Tagging beschränkt. Wenn Sie einem Benutzer beispielsweise mithilfe der `AdministratorAccess` AWS verwalteten Richtlinie vollständige Administratorrechte gewähren, wird dieser Zugriff durch diese Richtlinien nicht eingeschränkt. Weitere Informationen dazu, wie Berechtigungen festgelegt werden, wenn mehrere Richtlinien beteiligt sind, finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird.

IAM-Tutorial: Zulassen, dass Ihre Benutzer ihre eigenen Anmeldeinformationen und MFA-Einstellungen konfigurieren können

Auf der Seite Sicherheitsanmeldedaten können Sie Ihren Benutzern erlauben, ihre eigenen Geräte und Anmeldeinformationen mit Multi-Faktor-Authentifizierung (MFA) zu verwalten. Mit AWS Management Console können Sie Anmeldeinformationen (Zugriffsschlüssel, Passwörter, Signaturzertifikate und öffentliche SSH-Schlüssel) konfigurieren, nicht benötigte Anmeldeinformationen löschen oder deaktivieren und MFA-Geräte für Ihre Benutzer aktivieren. Dies ist für eine kleine Anzahl von Benutzern nützlich, diese Aufgabe kann jedoch schnell zeitaufwändig werden, wenn die Anzahl der Benutzer zunimmt. Ziel dieses Tutorials ist es, Ihnen zu zeigen, wie Sie diese bewährten Methoden umsetzen, ohne Ihre Administratoren zu belasten.

Dieses Tutorial zeigt, wie Sie Benutzern den Zugriff auf AWS Dienste ermöglichen, jedoch nur, wenn sie sich mit MFA anmelden. Wenn sie nicht mit einem MFA-Gerät angemeldet sind, können Benutzer nicht auf andere Services zugreifen.

Dieser Workflow umfasst drei grundlegende Schritte.

Schritt 1: Erstellen einer Richtlinie zum Erzwingen der MFA-Anmeldung

Erstellen einer vom Kunden verwalteten Richtlinie, die alle Aktionen verbietet außer die wenigen IAM-Aktionen. Diese Ausnahmen ermöglichen es Benutzern, ihre eigenen Anmeldeinformationen zu ändern und ihre MFA-Geräte auf der Seite Sicherheitsanmeldeinformationen zu verwalten. Weitere Informationen zum Zugriff auf diese Seite finden Sie unter Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole).

Schritt 2: Zuweisen von Richtlinien zu Ihrer Testgruppe

Erstellen Sie eine Gruppe, deren Mitglieder vollen Zugriff auf alle Amazon EC2-Aktionen haben, wenn sie sich mit MFA anmelden. Um eine solche Benutzergruppe zu erstellen, fügen Sie sowohl die aufgerufene AWS verwaltete Richtlinie als `AmazonEC2FullAccess` auch die vom Kunden verwaltete Richtlinie an, die Sie im ersten Schritt erstellt haben.

Schritt 3: Testen des Benutzerzugriffs

Melden Sie sich als Testbenutzer an, um zu überprüfen, ob der Zugriff auf Amazon EC2 blockiert ist, bis der Benutzer ein MFA-Gerät erstellt. Der Benutzer kann sich dann mit diesem Gerät anmelden.

Voraussetzungen

Um die Schritte in dieser praktischen Anleitung auszuführen, müssen Sie bereits über Folgendes verfügen:

- Und AWS-Konto bei der Sie sich als IAM-Benutzer mit Administratorrechten anmelden können.
- Ihre Konto-ID, die Sie in Schritt 1 in die Richtlinie eingeben.

  Um Ihre Konto-ID-Nummer zu finden, wählen Sie auf der Navigationsleiste oben auf der Seite die Option Support aus und klicken Sie dann auf Support Center. Sie finden Ihre Konto-ID im Menü Support dieser Seite.
- Ein virtuelles (softwarebasiertes) MFA-Gerät, FIDO-Sicherheitsschlüssel oder hardwarebasiertes MFA-Gerät.
- Ein IAM-Testbenutzer, der ein Mitglied einer Gruppe wie folgt ist:

Benutzer erstellen		Erstellen und Konfigurieren von Benutzergruppen		
Benutzername	Andere Anweisungen	Benutzergruppenname	Hinzufügen des Benutzers als Mitglied	Andere Anweisungen
MFAUser	Wählen Sie nur die Option für Enable console access – optional (Konsolenzugriff aktivieren – optional aus und weisen Sie ein Passwort zu.	EC2MFA	MFAUser	Ordnen Sie dieser Gruppe KEINE Richtlinien zu und erteilen Sie anderweitig Berechtigungen.

Schritt 1: Erstellen einer Richtlinie zum Erzwingen der MFA-Anmeldung

Sie beginnen mit dem Erstellen einer vom Kunden verwalteten IAM-Richtlinie, die alle Berechtigungen verweigert, mit Ausnahme derer, die erforderlich sind, damit IAM-Benutzer ihre eigenen Anmeldeinformationen und MFA-Geräte verwalten können.

1. Melden Sie sich als Benutzer mit Administratoranmeldedaten bei der AWS Management Console an. Melden Sie sich nicht mit Ihren Root-Benutzer des AWS-Kontos Anmeldeinformationen an, um sich an die Best Practices für IAM zu halten.

   > **Important**
   >
   > Bewährte IAM-Methoden empfehlen, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um mit temporären Anmeldeinformationen zuzugreifen AWS , anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen zu verwenden.

2. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

3. Wählen Sie im Navigationsbereich Policies (Richtlinien) und dann Create policy (Richtlinie erstellen).

4. Wählen Sie die Registerkarte JSON aus und kopieren Sie den Text aus dem folgenden JSON-Richtliniendokument: AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.
5. Fügen Sie den folgenden Text in das JSON-Eingabefeld ein. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Next (Weiter) aus.

   > **Note**
   >
   > Sie können jederzeit zwischen den Optionen Visual-Editor und JSON wechseln. Die Richtlinie oben enthält jedoch das `NotAction`-Element, welches im visuellen Editor nicht unterstützt wird. Für diese Richtlinie wird eine Benachrichtigung auf der Registerkarte Visual-Editor (Visueller Editor) angezeigt. Kehren Sie zu JSON zurück, um weiter mit dieser Richtlinie zu arbeiten.
   > Diese Beispielrichtlinie erlaubt es Benutzern nicht, ein Passwort zurückzusetzen, wenn sie sich zum ersten Mal bei AWS Management Console anmelden. Wir empfehlen, dass Sie neuen Benutzern keine Berechtigungen erteilen, bis sie sich angemeldet haben.

6. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) als Richtliniennamen **`Force_MFA`** ein. Geben Sie für die Richtlinienbeschreibung im Abschnitt Tags **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`** ein. Optional können Sie Tag-Schlüssel-Wert-Paare zur vom Kunden verwalteten Richtlinie hinzufügen. Überprüfen Sie die von ihrer Richtlinie erteilten Berechtigungen und wählen Sie dann zum Speichern Ihrer Arbeit Create policy (Richtlinie erstellen) aus.

   Die neue Richtlinie wird in der Liste der verwalteten Richtlinien angezeigt und ist bereit.

Schritt 2: Zuweisen von Richtlinien zu Ihrer Testgruppe

Als Nächstes ordnen Sie der Test-IAM-Benutzergruppe zwei Richtlinien zu, die für die Erteilung der MFA-geschützten Berechtigungen verwendet werden.

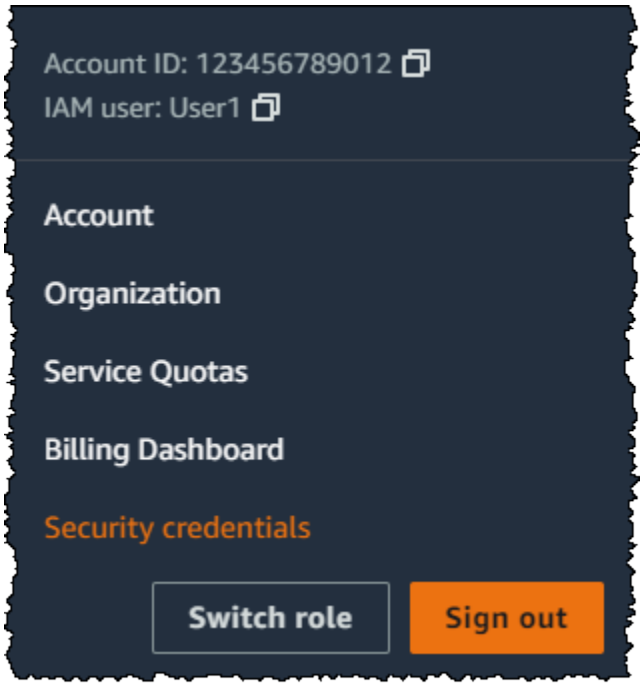
1. Klicken Sie im Navigationsbereich auf Groups oder Users.
2. Geben Sie **`EC2MFA`** in das Textfeld ein und wählen Sie dann den Gruppennamen (nicht das Kontrollkästchen) in der Liste.

3. Wählen Sie die Registerkarte Permissions (Berechtigungen), wählen Sie Add permissions (Berechtigungen hinzufügen) und wählen Sie dann Attach policies (Richtlinien anhängen).
4. Geben Sie auf der Seite Berechtigungsrichtlinien der EC2MFA-Gruppe zuordnen in das Suchfeld **`EC2Full`** ein. Aktivieren Sie dann das Kontrollkästchen neben AmazonEC2 FullAccess in der Liste. Speichern Sie Ihre Änderungen noch nicht.
5. Geben Sie in das Suchfeld **`Force`**Force ein und aktivieren Sie das Kontrollkästchen neben Force_MFA in der Liste.
6. Wählen Sie Attach Policies (Richtlinien hinzufügen).

Schritt 3: Testen des Benutzerzugriffs

In diesem Teil des Tutorials melden Sie sich als Testbenutzer an und überprüfen, ob die Richtlinie wie vorgesehen funktioniert.

1. Melden Sie sich **`MFAUser`** mit dem Passwort, das Sie im vorherigen Abschnitt vergeben haben, bei Ihrem AWS-Konto AS an. Verwenden Sie die URL: `https://`*`<alias or account ID number>`*`.signin.aws.amazon.com/console`
2. Wählen Sie EC2, um die Amazon EC2-Konsole zu öffnen und zu überprüfen, dass der Benutzer hat keine Berechtigungen für das Durchführen von Aktivitäten hat.
3. Wählen Sie auf der Navigationsleiste rechts oben den `MFAUser`-Benutzernamen und Security Credentials (Sicherheitsanmeldeinformationen).



4. Fügen Sie nun ein MFA-Gerät hinzu. Wählen Sie im Abschnitt Multi-Factor Authentication (MFA) die Option Assign MFA device (MFA-Gerät zuweisen).

   > **Note**
   >
   > Sie könnten eine Fehlermeldung erhalten, die darauf hinweist, dass Sie für die Ausführung von `iam:DeleteVirtualMFADevice` nicht autorisiert sind. Dies kann passieren, wenn jemand zuvor damit begonnen hat, ein virtuelles MFA-Gerät zu diesem Benutzern zuzuweisen und den Prozess dann abgebrochen hat. Damit Sie fortfahren können, müssen Sie oder ein anderer Administrator das vorhandene nicht zugewiesene, virtuelle MFA-Gerät des Benutzers löschen. Weitere Informationen finden Sie unter Ich bin nicht berechtigt, Folgendes auszuführen: iam: MFADevice DeleteVirtual.

5. Für dieses Tutorial verwenden wir ein virtuelles (softwarebasiertes) MFA-Gerät, wie z. B. die Google Authenticator-App auf einem Mobiltelefon. Wählen Sie die Authenticator-App und klicken Sie dann auf Next (Weiter).

   IAM generiert Konfigurationsinformationen für das virtuelle MFA-Gerät und zeigt diese einschließlich eines QR-Codes an. Dieser Code ist eine grafische Darstellung des geheimen Konfigurationsschlüssels, der für die manuelle Eingabe auf Geräte zur Verfügung steht, die keine QR-Codes unterstützen.

6. Öffnen Sie Ihre virtuelle MFA-App. (Eine Liste der Anwendungen, die Sie zum Hosten von virtuellen MFA-Geräten verwenden können, finden Sie unter Virtuelle MFA-Anwendungen.) Wenn die virtuelle MFA-App mehrere Konten (mehrere virtuelle MFA-Geräte) unterstützt, wählen Sie die Option zum Erstellen eines neuen Kontos (eines neues virtuellen MFA-Geräts).

7. Stellen Sie fest, ob die MFA-App QR-Codes unterstützt, und führen Sie dann einen der folgenden Schritte aus:

   - Wählen Sie im Assistenten Show QR-Code (QR-Code anzeigen). Verwenden Sie dann die App, um den QR-Code zu scannen. Sie können beispielsweise das Kamerasymbol oder eine Anwendung wie z. B. Scan Code (Code scannen) auswählen und dann mit der Kamera des Geräts den Code scannen.
   - Wählen Sie im Assistenten zum Einrichten des Geräts die Option Show secret key (Geheimen Schlüssel anzeigen) aus und geben Sie dann den geheimen Schlüssel in Ihre MFA-App ein.

   Wenn Sie fertig sind, beginnt das virtuelle MFA-Gerät, einmalige Passwörter zu generieren.

8. Geben Sie im Assistenten zum Einrichten eines Geräts im Feld Authentication Code 1 das aktuell am virtuellen MFA-Gerät angezeigte einmalige Passwort ein. Wählen Sie die Option Register MFA (MFA registrieren).

   > **Important**
   >
   > Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes generieren und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit dem Benutzer verknüpft. Allerdings ist das MFA-Gerät nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. In diesem Fall können Sie das Gerät neu synchronisieren.

   Das virtuelle MFA-Gerät ist jetzt einsatzbereit. AWS
9. Melden Sie sich bei der Konsole ab und anschließend erneut als **`MFAUser`** an. Diesmal AWS werden Sie aufgefordert, einen MFA-Code von Ihrem Telefon einzugeben. Wenn Sie ihn erhalten, geben Sie den Code in das Feld ein und wählen Sie dann Submit (Absenden).
10. Wählen Sie EC2, um die Amazon EC2-Konsole erneut zu öffnen. Beachten Sie, dass Sie jetzt alle Informationen sehen und alle gewünschten Aktionen ausführen können. Wenn Sie als dieser Benutzer zu einer anderen Konsole wechseln, erhalten Sie Meldungen, die besagen, dass der Zugriff verweigert wurde. Der Grund dafür ist, dass die Richtlinien in diesem Tutorial nur Zugriff auf Amazon EC2 gewähren.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter den folgenden Themen:

- Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS
- Aktivierung von MFA-Geräten für Benutzer in AWS
- Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite

IAM-Identitäten (Benutzer, Gruppen und Rollen)

> **Tip**
>
> Haben Sie Probleme bei der Anmeldung? AWS Stellen Sie sicher, dass Sie sich auf der richtigen Anmeldeseite.
>
> - Um sich als Root-Benutzer des AWS-Kontos (Kontoinhaber) anzumelden, verwenden Sie die Anmeldeinformationen, die Sie bei der Erstellung des eingerichtet haben AWS-Konto.
> - Zum Anmelden als IAM-Benutzer verwenden Sie die Anmeldeinformationen, die Ihnen Ihr Kontoadministrator zur Anmeldung in AWS gegeben hat.
> - 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 Access-Portal.
>
> Tutorials zur Anmeldung finden Sie unter So melden Sie sich in AWS an im AWS-Anmeldung -Benutzerhandbuch.

> **Note**
>
> Wenn Sie Support anfordern möchten, verwenden Sie nicht den Feedback-Link auf dieser Seite. Feedback, das Sie eingeben, geht an das AWS Dokumentationsteam, nicht an den AWS Support. Wählen Sie stattdessen oben auf dieser Seite den Link Kontakt aus. Dort finden Sie Links zu Ressourcen, mit denen Sie die Unterstützung erhalten, die Sie benötigen.

Der Root-Benutzer des AWS-Kontos oder ein Administratorbenutzer für das Konto kann IAM-Identitäten erstellen. Eine IAM-Identität ermöglicht den Zugriff auf ein AWS-Konto. Eine IAM-Benutzergruppe ist eine Sammlung von IAM-Benutzern, die als eine Einheit verwaltet werden. Eine IAM-Identität stellt einen menschlichen Benutzer oder einen programmgesteuerten Workload dar, der authentifiziert und anschließend zur Durchführung von Aktionen in AWS autorisiert werden kann. Jede IAM-Identität kann einer oder mehreren Richtlinien zugeordnet werden. Richtlinien legen fest,

welche Aktionen ein Benutzer, eine Rolle oder ein Mitglied einer Benutzergruppe auf welchen AWS Ressourcen und unter welchen Bedingungen ausführen kann.

AWS-Konto Root-Benutzer

Wenn Sie zum ersten Mal 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.

> **Important**
>
> 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 Stammbenutzer-Anmeldeinformationen erfordern.

IAM-Benutzer

Ein IAM-Benutzer ist eine Identität innerhalb von Ihnen AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen die bewährten Methoden, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bevor Sie Zugriffsschlüssel erstellen, prüfen Sie die Alternativen zu Langzeit-Zugriffsschlüsseln. Wenn Sie bestimmte Anwendungsfälle haben, die Zugriffsschlüssel erfordern, empfehlen wir Ihnen, die Zugriffsschlüssel bei Bedarf zu aktualisieren. Weitere Informationen finden Sie unter Aktualisieren Sie Zugriffsschlüssel für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern. Informationen zum Hinzufügen von IAM-Benutzern zu Ihrem System finden Sie AWS-Konto unter. Erstellen eines IAM-Benutzers in Ihrem AWS-Konto

> **Note**
>
> Im Sinne bewährter Sicherheitsmethoden wird empfohlen, den Zugriff auf Ihre Ressourcen über einen Identitätsverbund zu ermöglichen, anstatt IAM-Benutzer zu erstellen.

Informationen zu bestimmten Situationen, in denen ein IAM-Benutzer erforderlich ist, finden Sie unter Wann sollte ein IAM-Benutzer (anstelle einer Rolle) erstellt werden?.

IAM-Benutzergruppen

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht mit einer 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 können beispielsweise eine Gruppe mit dem Namen IAMPublishers erstellen und dieser Gruppe die für Workloads üblichen Berechtigungen zuweisen.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb Ihres Unternehmens, für AWS-Konto die bestimmte Berechtigungen gelten. Es ähnelt einem IAM-Benutzer, ist jedoch keiner bestimmten Person zugeordnet. 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.

IAM-Rollen mit temporären Anmeldeinformationen werden in folgenden Situationen verwendet:

- 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. In einigen AWS-Services können Sie jedoch eine Richtlinie direkt an eine Ressource anfügen (anstatt eine Rolle als Proxy zu verwenden). Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für kontenübergreifenden Zugriff finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.

- 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.
- Anwendungen, die auf Amazon EC2 ausgeführt werden — 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.

Temporäre Anmeldeinformationen in IAM

Als bewährte Methode sollten Sie temporäre Anmeldeinformationen sowohl für menschliche Benutzer als auch für Workloads verwenden. Temporäre Anmeldeinformationen werden hauptsächlich mit IAM-Rollen verwendet, es gibt aber auch andere Verwendungsmöglichkeiten. Sie können temporäre Anmeldeinformationen mit eingeschränkteren Berechtigungen als Ihr standardmäßiger IAM-Benutzer anfordern. Dadurch wird verhindert, dass Sie versehentlich Aufgaben ausführen, die den eingeschränkteren Berechtigungen zufolge nicht zulässig sind. Ein Vorteil der temporären Anmeldeinformationen besteht darin, dass sie automatisch nach einem bestimmten Zeitraum ablaufen. Sie kontrollieren die Gültigkeitsdauer der Anmeldeinformationen.

Wann sollten IAM-Identity-Center-Benutzer verwendet werden?

Wir empfehlen allen menschlichen Benutzern, IAM Identity Center für den Zugriff auf AWS Ressourcen zu verwenden. IAM Identity Center ermöglicht deutliche Verbesserungen gegenüber dem Zugriff auf AWS Ressourcen als IAM-Benutzer. IAM Identity Center bietet Folgendes:

- Zentraler Satz von Identitäten und Zuweisungen
- Zugriff auf Konten im gesamten Unternehmen AWS
- Verbindung zu Ihrem bestehenden Identitätsanbieter
- Temporäre Anmeldeinformationen
- Multi-Faktor-Authentifizierung (MFA)
- Self-Service-MFA-Konfiguration für Endbenutzer
- Administrative Durchsetzung der Verwendung von MFA
- Single Sign-On für alle Berechtigungen AWS-Konto

Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

Erstellen eines IAM-Benutzers (anstatt eine Rolle)

Wir empfehlen, IAM-Benutzer nur für Anwendungsfälle zu verwenden, die von Verbundbenutzern nicht unterstützt werden. Einige der Anwendungsfälle umfassen Folgendes:

- Workloads, die IAM-Rollen nicht verwenden können – Sie könnten einen Workload von einem Speicherort ausführen, der auf AWS zugreifen muss. In einigen Situationen können Sie IAM-Rollen nicht verwenden, um temporäre Anmeldeinformationen bereitzustellen, z. B. für Plugins. WordPress Verwenden Sie in diesen Situationen langfristige IAM-Benutzerzugriffsschlüssel für diesen Workload, um sich bei AWS zu authentifizieren.
- AWS Drittanbieter-Clients — Wenn Sie Tools verwenden, die den Zugriff mit IAM Identity Center nicht unterstützen, z. B. AWS Drittanbieter-Clients oder Anbieter, die nicht auf gehostet werden AWS, verwenden Sie langfristige IAM-Benutzerzugriffsschlüssel.
- AWS CodeCommit Zugriff — Wenn Sie Ihren Code CodeCommit zum Speichern verwenden, können Sie einen IAM-Benutzer mit SSH-Schlüsseln oder dienstspezifischen Anmeldeinformationen verwenden, um sich bei Ihren Repositorys CodeCommit zu authentifizieren. Wir empfehlen Ihnen, dies zusätzlich zu einem Benutzer im IAM Identity Center für die normale Authentifizierung zu verwenden. Benutzer in IAM Identity Center sind die Personen in Ihrer Belegschaft, die Zugriff auf Ihre oder Ihre AWS-Konten Cloud-Anwendungen benötigen. Um Benutzern Zugriff auf Ihre CodeCommit Repositorys zu gewähren, ohne IAM-Benutzer zu konfigurieren, können Sie das Hilfsprogramm konfigurieren. git-remote-codecommit Weitere Informationen zu IAM und CodeCommit finden Sie unter. Verwenden von IAM mit CodeCommit: Git-Anmeldeinformationen, SSH-Schlüsseln und AWS Zugriffsschlüsseln Weitere Informationen zur Konfiguration des git-remote-codecommit Dienstprogramms finden Sie im AWS CodeCommit Benutzerhandbuch unter Herstellen einer Verbindung zu AWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen.
- Zugriff auf Amazon Keyspaces (für Apache Cassandra) – In einer Situation, in der Sie Benutzer im IAM Identity Center nicht verwenden können, z. B. zu Testzwecken für die Cassandra-Kompatibilität, können Sie einen IAM-Benutzer mit dienstspezifischen Anmeldeinformationen verwenden, um sich bei Amazon Keyspaces zu authentifizieren. Benutzer in IAM Identity Center sind die Personen in Ihrer Belegschaft, die Zugriff auf Ihre AWS-Konten oder Ihre Cloud-Anwendungen benötigen. Sie können auch mithilfe temporärer Anmeldeinformationen eine Verbindung zu Amazon Keyspaces herstellen. Weitere Informationen finden Sie unter Verwendung temporärer Anmeldeinformationen für die Verbindung zu Amazon Keyspaces mithilfe einer IAM-Rolle und des SigV4-Plugins im Amazon Keyspaces (für Apache Cassandra)-Entwicklerhandbuch.

- Notfallzugriff – In einer Situation, in der Sie keinen Zugriff auf Ihren Identitätsanbieter haben und in Ihrem AWS-Konto Maßnahmen ergreifen müssen. Die Einrichtung von IAM-Benutzern für den Notfallzugriff kann Teil Ihres Resilienzplans sein. Wir empfehlen Ihnen, die Anmeldeinformationen für den Notfallbenutzer genau zu kontrollieren und sie mit Multi-Faktor-Authentifizierung (MFA) zu sichern.

Erstellen einer IAM-Rolle (anstatt eines Benutzers)

Erstellen Sie in folgenden Fällen eine IAM-Rolle:

Sie erstellen eine Anwendung, die auf einer Amazon Elastic Compute Cloud (Amazon EC2) -Instance läuft und an AWS die diese Anwendung Anfragen sendet.

> Wir raten davon ab, einen IAM-Benutzer zu erstellen und die Benutzer-Anmeldeinformationen an die Anwendung zu übermitteln oder die Anmeldeinformationen in die Anwendung einzubetten. Erstellen Sie stattdessen eine IAM-Rolle, die Sie an die EC2-Instance anfügen, um den auf der Instance ausgeführten Anwendungen temporäre Sicherheitsanmeldeinformationen bereitzustellen. Wenn eine Anwendung diese Anmeldeinformationen verwendet AWS, kann sie alle Operationen ausführen, die gemäß den mit der Rolle verknüpften Richtlinien zulässig sind. Details hierzu finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden.

Sie erstellen eine Anwendung, die auf einem Mobiltelefon ausgeführt wird und Anforderungen an AWS sendet.

> Wir raten davon ab, einen IAM-Benutzer zu erstellen und den Zugriffsschlüssel des Benutzers mit der Anwendung zu verteilen. Verwenden Sie stattdessen einen Identitätsanbieter wie Login with Amazon, Amazon Cognito, Facebook oder Google, um Benutzer zu authentifizieren und die Benutzer einer IAM-Rolle zuzuordnen. Die Anwendung kann die Rolle verwenden, um temporäre Sicherheitsanmeldeinformationen zu erhalten, die über die Richtlinien verfügen, die den Berechtigungen entsprechend zur Rolle angefügt sind. Weitere Informationen finden Sie hier:
>
> - Amazon-Cognito-Benutzerhandbuch
> - OIDC-Föderation

Die Benutzer in Ihrem Unternehmen sind in Ihrem Unternehmensnetzwerk authentifiziert und möchten die Nutzung nutzen können, AWS ohne sich erneut anmelden zu müssen. Das heißt, Sie möchten Benutzern die Möglichkeit geben, sich zu verbinden. AWS

> Wir raten davon ab, IAM-Benutzer zu erstellen. Konfigurieren Sie eine Verbundbeziehung zwischen Ihrem Unternehmensidentitätssystem und. AWS Es gibt zwei Methoden dafür:
>
> - Wenn das Identitätssystem Ihres Unternehmens mit SAML 2.0 kompatibel ist, können Sie eine Vertrauensstellung zwischen dem Identitätssystem Ihres Unternehmens und AWS herstellen. Weitere Informationen finden Sie unter SAML 2.0-Verbund.
> - Erstellen und verwenden Sie einen benutzerdefinierten Proxyserver, der Benutzeridentitäten aus dem Unternehmen in IAM-Rollen übersetzt, die temporäre Sicherheitsanmeldedaten bereitstellen. AWS Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

Vergleichen Sie Root-Benutzer des AWS-Kontos Anmeldeinformationen und IAM-Benutzeranmeldedaten

Der Root-Benutzer ist der Kontoinhaber und wird erstellt, wenn der erstellt AWS-Konto wird. Andere Benutzertypen, einschließlich IAM-Benutzer, und AWS IAM Identity Center Benutzer werden vom Root-Benutzer oder einem Administrator für das Konto erstellt. Alle AWS Benutzer verfügen über Sicherheitsanmeldedaten.

Anmeldeinformationen des Stammbenutzers

Diese Anmeldeinformationen des Kontoinhabers ermöglichen vollständigen Zugriff auf alle Ressourcen des Kontos. Sie können keine IAM-Richtlinien verwenden, um dem Root-Benutzer den Zugriff auf Ressourcen explizit zu verweigern. Sie können nur eine AWS Organizations Service Control Policy (SCP) verwenden, um die Berechtigungen des Root-Benutzers eines Mitgliedskontos einzuschränken. Aus diesem Grund empfehlen wir, in IAM Identity Center einen Administratorbenutzer zu erstellen, den Sie für alltägliche AWS Aufgaben verwenden können. Sichern Sie dann die Anmeldeinformationen des Root-Benutzers und verwenden Sie sie nur für die wenigen Aufgaben der Konto- und Service-Verwaltung, für die Sie sich als Root-Benutzer anmelden müssen. Eine Liste dieser Aufgaben finden Sie unter Aufgaben, die Stammbenutzer-Anmeldeinformationen erfordern. Informationen zum Einrichten eines Administrators für den täglichen Gebrauch in IAM Identity Center finden Sie unter Erste Schritte im IAM-Identity-Center-Benutzerhandbuch.

IAM-Anmeldeinformationen

Ein IAM-Benutzer ist eine von Ihnen erstellte Entität AWS , die die Person oder den Dienst darstellt, der den IAM-Benutzer für die Interaktion mit Ressourcen verwendet. AWS Bei diesen Benutzern handelt es sich um Identitäten innerhalb von Ihnen AWS-Konto , die über spezifische benutzerdefinierte Berechtigungen verfügen. Sie können beispielsweise IAM-Benutzer erstellen und ihnen Berechtigungen zum Erstellen eines Verzeichnisses im IAM Identity Center gewähren. IAM-Benutzer verfügen über langfristige Anmeldeinformationen, mit denen sie über die oder programmgesteuert AWS über die AWS Management Console APIs oder zugreifen können. AWS CLI AWS step-by-step Anweisungen dazu, wie sich IAM-Benutzer bei der anmelden AWS Management Console, finden Sie unter AWS Management Console Als IAM-Benutzer anmelden im Anmelde-Benutzerhandbuch.AWS

Im Allgemeinen empfehlen wir, das Erstellen von IAM-Benutzern zu vermeiden, da diese über langfristige Anmeldeinformationen wie einen Benutzernamen und ein Kennwort verfügen. Erfordern Sie stattdessen, dass menschliche Benutzer beim Zugriff temporäre Anmeldeinformationen verwenden. AWS Sie können einen Identitätsanbieter für Ihre menschlichen Benutzer verwenden, um Verbundzugriff zu gewähren, AWS-Konten indem Sie IAM-Rollen übernehmen, die temporäre Anmeldeinformationen bereitstellen. Für eine zentrale Zugriffsverwaltung empfehlen wir Ihnen die Verwendung von IAM Identity Center, um den Zugriff auf Ihre Konten und die Berechtigungen innerhalb dieser Konten zu verwalten. Sie können Ihre Benutzeridentitäten mit IAM Identity Center verwalten oder Zugriffsberechtigungen für Benutzeridentitäten in IAM Identity Center von einem externen Identitätsanbieter verwalten. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im IAM-Identity-Center-Benutzerhandbuch.

Root-Benutzer des AWS-Kontos

Wenn Sie zum ersten Mal ein Amazon Web Services (AWS) -Konto erstellen, beginnen Sie mit einer einzigen Anmeldeidentität, die vollständigen Zugriff auf alle AWS Dienste und Ressourcen im Konto hat. Diese Identität wird als Root-Benutzer des AWS Kontos bezeichnet. Der Zugriff erfolgt, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben.

> **Important**
>
> Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden und empfehlen, dass Sie die bewährten Methoden für Root-Benutzer für Ihr AWS-Konto befolgen. 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 Stammbenutzer-Anmeldeinformationen erfordern.

In den folgenden Themen werden die mit dem Root-Benutzer verbundenen Verwaltungsaufgaben detailliert beschrieben.

Aufgaben

- Aktivieren Sie die Multi-Faktor-Authentifizierung für Ihre Root-Benutzer des AWS-Kontos (Konsole)
- Ändern Sie das Passwort für Root-Benutzer des AWS-Kontos
- Zurücksetzen eines verlorenen oder vergessenen Stammbenutzer-Passworts
- Erstellen von Zugriffsschlüsseln für den Stammbenutzer
- Löschen von Zugriffsschlüsseln für den Stammbenutzer
- Aufgaben, die Stammbenutzer-Anmeldeinformationen erfordern
- Fehlerbehebung bei Problemen mit dem Root-Benutzer
- Ähnliche Informationen

Aktivieren Sie die Multi-Faktor-Authentifizierung für Ihre Root-Benutzer des AWS-Kontos (Konsole)

Die Multi-Faktor-Authentifizierung (MFA) ist ein einfacher und effektiver Mechanismus zur Verbesserung Ihrer Sicherheit. Der erste Faktor — Ihr Passwort — ist ein Geheimnis, das Sie sich merken, auch Wissensfaktor genannt. Andere Faktoren können Besitzfaktoren (etwas, das Sie besitzen, z. B. ein Sicherheitsschlüssel) oder Inhärenzfaktoren (etwas, das Sie sind, z. B. ein biometrischer Scan) sein. Um die Sicherheit zu erhöhen, empfehlen wir dringend, die Multi-Faktor-Authentifizierung (MFA) zu konfigurieren, um Ihre AWS Ressourcen zu schützen.

Sie können MFA für die Root-Benutzer des AWS-Kontos und IAM-Benutzer aktivieren. Wenn Sie MFA für den Root-Benutzer aktivieren, wirkt sich dies nur auf die Root-Benutzeranmeldedaten aus. Weitere Informationen zur Aktivierung von MFA für Ihre IAM-Benutzer finden Sie unter Aktivieren von MFA-Geräten für IAM-Benutzer in. AWS

Bevor Sie MFA für Ihren Root-Benutzer aktivieren, überprüfen und aktualisieren Sie Ihre Kontoeinstellungen und Kontaktinformationen, um sicherzustellen, dass Sie Zugriff auf die E-Mail-Adresse und Telefonnummer haben. Wenn Ihr MFA-Gerät verloren geht, gestohlen wird oder nicht

funktioniert, können Sie sich immer noch als Stammbenutzer anmelden, indem Sie Ihre Identität mit dieser E-Mail und Telefonnummer verifizieren. Weitere Informationen zum Anmelden mit alternativen Authentifizierungsmethoden finden Sie unter Was passiert, wenn ein MFA-Gerät verloren geht oder nicht funktioniert?. Wenn Sie dieses Feature deaktivieren möchten, wenden Sie sich an AWS Support.

Themen

Verfügbare MFA-Typen für einen Root-Benutzer

AWS unterstützt die folgenden MFA-Typen für Ihren Root-Benutzer: Hauptschlüssel und Sicherheitsschlüssel, virtuelle Authentifikatoranwendungen und Hardware-TOTP-Token.

Hauptschlüssel und Sicherheitsschlüssel

AWS Identity and Access Management unterstützt Hauptschlüssel und Sicherheitsschlüssel für MFA. Basierend auf den FIDO-Standards verwenden Hauptschlüssel Kryptografie mit öffentlichen Schlüsseln, um eine starke, gegen Phishing resistente Authentifizierung zu gewährleisten, die sicherer ist als Passwörter. AWS unterstützt zwei Arten von Hauptschlüsseln: gerätegebundene Hauptschlüssel (Sicherheitsschlüssel) und synchronisierte Hauptschlüssel.

- Sicherheitsschlüssel: Dabei handelt es sich um physische Geräte, wie z. B. a YubiKey, die als zweiter Faktor für die Authentifizierung verwendet werden.
- Synchronisierte Hauptschlüssel: Diese verwenden Anmeldeinformationsmanager von Anbietern wie Google, Apple, Microsoft-Konten und Drittanbieterdiensten wie 1Password, Dashlane und Bitwarden als zweiten Faktor.

Sie können integrierte biometrische Authentifikatoren wie Touch ID auf Apple MacBooks und Windows Hello-Gesichtserkennung auf PCs verwenden, um Ihren Anmeldeinformationsmanager zu entsperren und sich dort anzumelden. AWS Hauptschlüssel werden bei dem von Ihnen ausgewählten Anbieter anhand Ihres Fingerabdrucks, Ihres Gesichts oder Ihrer Geräte-PIN erstellt. Sie können

Hauptschlüssel auf Ihren Geräten synchronisieren, um die Anmeldung zu erleichtern und so die Benutzerfreundlichkeit und Wiederherstellbarkeit zu AWS verbessern.

Die FIDO Alliance führt eine Liste aller FIDO-zertifizierten Produkte, die mit den FIDO-Spezifikationen kompatibel sind. Ein einziger Hauptschlüssel oder Sicherheitsschlüssel unterstützt mehrere Root-Benutzerkonten und IAM-Benutzer. Weitere Informationen zur Aktivierung von Hauptschlüsseln und Sicherheitsschlüsseln finden Sie unter. Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für die Root-Benutzer des AWS-Kontos (Konsole)

Anwendungen für virtuelle Authentifikatoren

Eine virtuelle Authentifizierungsanwendung wird auf einem Telefon oder einem anderen Gerät ausgeführt und emuliert ein physisches Gerät. Virtuelle Authentifizierungs-Apps implementieren den Algorithmus für ein zeitgesteuertes Einmalpasswort (TOTP) und unterstützen mehrere Token auf einem einzigen Gerät. Der Benutzer muss einen gültigen Code vom Gerät eingeben, wenn er bei der Anmeldung dazu aufgefordert wird. Jedes einem Benutzer zugewiesene Token muss eindeutig sein. Ein Benutzer kann zur Authentifizierung keinen Code aus dem Token eines anderen Benutzers eingeben.

Wir empfehlen die Verwendung eines virtuellen MFA-Geräts beim Warten auf die Genehmigung für den Hardware-Kauf oder während Sie warten, bis Ihre Hardware eintrifft. Eine Liste einiger unterstützter Apps, die Sie als virtuelle MFA-Geräte verwenden können, finden Sie unter Multi-Factor Authentication (MFA). Anweisungen zum Einrichten eines virtuellen MFA-Geräts mit finden Sie AWS unterAktivieren eines virtuellen MFA-Geräts für Ihren Root-Benutzer des AWS-Kontos (Konsole).

Hardware-TOTP-Token

Ein Hardwaregerät generiert einen sechsstelligen numerischen Code, der auf dem TOTP-Algorithmus (Time-Based One-Time Password) basiert. Der Benutzer muss während der Anmeldung auf einer zweiten Webseite einen gültigen Code von dem Gerät eingeben. Jedes MFA-Gerät, das einem Benutzer zugeordnet ist, muss eindeutig sein. Ein Benutzer kann keinen Code von einem MFA-Gerät eines anderen Benutzers eingeben, um sich zu authentifizieren. Informationen zu unterstützten Hardware-MFA-Geräten finden Sie unter Multi-Factor Authentication (MFA). Anweisungen zur Einrichtung eines Hardware-TOTP-Tokens mit finden Sie unter. AWSAktivieren Sie ein Hardware-TOTP-Token für die Root-Benutzer des AWS-Kontos (Konsole)

Wenn Sie ein physisches MFA-Gerät verwenden möchten, empfehlen wir Ihnen, FIDO-Sicherheitsschlüssel als Alternative zu Hardware-TOTP-Geräten zu verwenden. FIDO-Sicherheitsschlüssel bieten den Vorteil, dass sie keine Batterien benötigen und Phishing-Angriffe

verhindern. Außerdem unterstützen sie mehrere Root- und IAM-Benutzer auf einem einzigen Gerät, um die Sicherheit zu erhöhen.

Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für die Root-Benutzer des AWS-Kontos (Konsole)

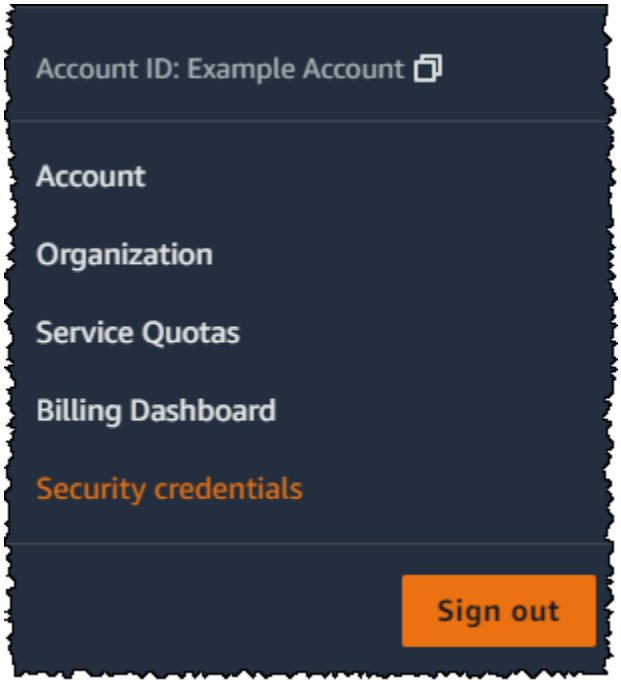
Sie können einen Hauptschlüssel für Ihren Root-Benutzer AWS Management Console nur über die API konfigurieren und aktivieren, nicht über die AWS CLI oder AWS .

Um einen Hauptschlüssel oder Sicherheitsschlüssel für Ihren Root-Benutzer (Konsole) zu aktivieren

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

   > **Note**
   >
   > Als Root-Benutzer können Sie sich nicht auf der Seite Sign in as IAM user (Als IAM-Benutzer anmelden) anmelden. Wenn Ihnen die Seite Sign in as IAM user (Als IAM-Benutzer anmelden) angezeigt wird, wählen Sie die Option Sign in using root user email (Mit Root-Benutzer-E-Mail anmelden) unten auf der Seite. Hilfe bei der Anmeldung als Root-Benutzer finden Sie im Benutzerhandbuch unter AWS Management Console Als AWS-Anmeldung Root-Benutzer anmelden.

2. Wählen Sie rechts in der Navigationsleiste Ihren Kontonamen und wählen Sie dann Security Credentials (Sicherheitsanmeldeinformationen). Sofern erforderlich, wählen Sie Continue to Security Credentials (Weiter zu Sicherheitsanmeldeinformationen).



3. Wählen Sie auf der Seite Meine Sicherheitsanmeldedaten Ihres Root-Benutzers unter Multi-Factor Authentication (MFA) die Option MFA-Gerät zuweisen aus.
4. Geben Sie auf der Seite MFA-Gerätename einen Gerätenamen ein, wählen Sie Hauptschlüssel oder Sicherheitsschlüssel und dann Weiter.
5. Richten Sie auf Gerät einrichten Ihren Hauptschlüssel ein. Erstellen Sie einen Hauptschlüssel mit biometrischen Daten wie Ihrem Gesicht oder Fingerabdruck, mit einer Geräte-PIN oder indem Sie den FIDO-Sicherheitsschlüssel in den USB-Anschluss Ihres Computers stecken und darauf tippen.
6. Folgen Sie den Anweisungen in Ihrem Browser, um einen Hauptschlüsselanbieter auszuwählen oder den Ort auszuwählen, an dem Sie Ihren Hauptschlüssel speichern möchten, um ihn auf Ihren Geräten zu verwenden.
7. Klicken Sie auf Weiter.

Sie haben jetzt Ihren Hauptschlüssel für die Verwendung mit registriert. AWS Wenn Sie sich das nächste Mal mit Ihren Root-Benutzeranmeldedaten anmelden, müssen Sie sich mit Ihrem Hauptschlüssel authentifizieren, um den Anmeldevorgang abzuschließen.

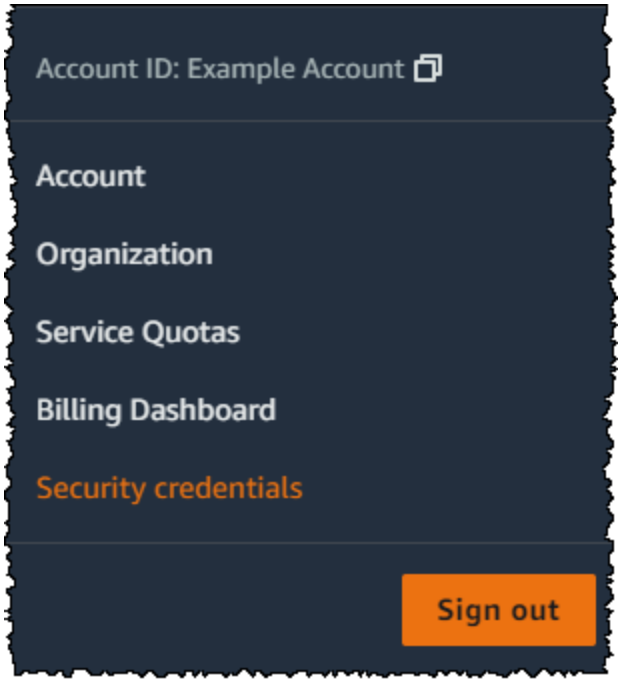
Hilfe zur Behebung von Problemen mit Ihrem FIDO-Sicherheitsschlüssel finden Sie unter. Fehlerbehebung von FIDO-Sicherheitsschlüsseln

Aktivieren eines virtuellen MFA-Geräts für Ihren Root-Benutzer des AWS-Kontos (Konsole)

Sie können das verwenden AWS Management Console , um ein virtuelles MFA-Gerät für Ihren Root-Benutzer zu konfigurieren und zu aktivieren. Um MFA-Geräte für zu aktivieren AWS-Konto, müssen Sie AWS mit Ihren Root-Benutzeranmeldedaten angemeldet sein.

So konfigurieren und aktivieren Sie ein virtuelles MFA-Gerät zur Verwendung mit dem (Konsole)

1. Melden Sie sich bei der AWS Management Console an.
2. Klicken Sie rechts auf der Navigationsleiste auf Ihren Kontonamen und wählen Sie Security Credentials (Sicherheits-Anmeldeinformationen) aus. Sofern erforderlich, wählen Sie Continue to Security Credentials (Weiter zu Sicherheitsanmeldeinformationen).



3. Wählen Sie im Abschnitt Multi-Factor Authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) die Option Assign MFA device (MFA-Gerät zuweisen).
4. Geben Sie im Assistenten einen Device name (Gerätenamen) ein, wählen Sie Authenticator app (Authenticator-App) und dann Next (Weiter).

   IAM generiert Konfigurationsinformationen für das virtuelle MFA-Gerät und zeigt diese einschließlich eines QR-Codes an. Dieser Code ist eine grafische Darstellung des geheimen Konfigurationsschlüssels, der für die manuelle Eingabe auf Geräte zur Verfügung steht, die keine QR-Codes unterstützen.

5. Öffnen Sie die virtuelle MFA-App auf dem Gerät.

   Wenn die virtuelle MFA-App mehrere virtuelle MFA-Geräte oder -Konten unterstützt, wählen Sie die Option zum Erstellen eines neuen virtuellen MFA-Geräts oder -Kontos.

6. Die einfachste Methode zum Konfigurieren der App ist die Verwendung der App zum Scannen des QR-Codes. Wenn es nicht möglich ist, den Code zu scannen, können Sie die Konfiguration manuell eingeben. Der von IAM generierte QR-Code und der geheime Konfigurationsschlüssel sind an Ihr Konto gebunden AWS-Konto und können nicht mit einem anderen Konto verwendet werden. Sie können jedoch zum Konfigurieren eines neuen MFA-Geräts für Ihr Konto wiederverwendet werden, falls Sie den Zugriff auf das ursprüngliche MFA-Gerät verlieren.

   - Um den QR-Code zur Konfiguration des virtuellen MFA-Geräts zu verwenden, wählen Sie im Assistenten Show QR code (QR-Code anzeigen). Folgen Sie dann den Anweisungen der App zum Scannen des Codes. Sie müssen beispielsweise das Kamerasymbol oder einen Tippbefehl wie z. B. Scan account barcode (Barcode des Kontos scannen) auswählen und dann mit der Kamera des Geräts den QR-Code scannen.
   - Wählen Sie im Assistenten zum Einrichten des Geräts die Option Show secret key (Geheimen Schlüssel anzeigen) aus und geben Sie dann den geheimen Schlüssel in Ihre MFA-App ein.

   > **Important**
   >
   > Erstellen Sie eine sichere Kopie des QR-Codes oder geheimen Zugriffsschlüssels oder stellen Sie sicher, dass Sie mehrere MFA-Geräte für Ihr Konto aktivieren. Sie können bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen bei Ihren Root-Benutzer des AWS-Kontos und IAM-Benutzern registrieren. Ein virtuelles MFA-Gerät ist möglicherweise nicht mehr verfügbar, wenn Sie beispielsweise das Smartphone verlieren, auf dem das virtuelle MFA-Gerät gehostet wird. Wenn dies geschieht und Sie sich ohne zusätzliche MFA-Geräte, die an den Benutzer angeschlossen sind, oder sogar über Wiederherstellen eines Stammbenutzer-MFA-Geräts nicht bei Ihrem Konto anmelden können, können Sie sich nicht bei Ihrem Konto anmelden und müssen den Kundendienst kontaktieren, um den MFA-Schutz für das Konto zu entfernen.

   Das Gerät beginnt, sechsstellige Zahlen zu generieren.

7. Geben Sie im Assistenten im Feld MFA Code 1 das aktuell am virtuellen MFA-Gerät angezeigte Einmalpasswort ein. Warten Sie bis zu 30 Sekunden, bis das Gerät ein neues einmaliges Passwort generiert. Geben Sie dann das zweite Einmalpasswort in das Feld MFA Code 2 ein. Wählen Sie Add MFA (MFA hinzufügen).

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes generieren und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit dem Benutzer verknüpft, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. In diesem Fall können Sie das Gerät neu synchronisieren.

Das Gerät ist einsatzbereit für. AWS Weitere Informationen zur Verwendung von MFA mit der AWS Management Console finden Sie unter Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Aktivieren Sie ein Hardware-TOTP-Token für die Root-Benutzer des AWS-Kontos (Konsole)

Sie können ein physisches MFA-Gerät für Ihren Root-Benutzer AWS Management Console nur über die AWS API konfigurieren und aktivieren, nicht über die AWS CLI oder.

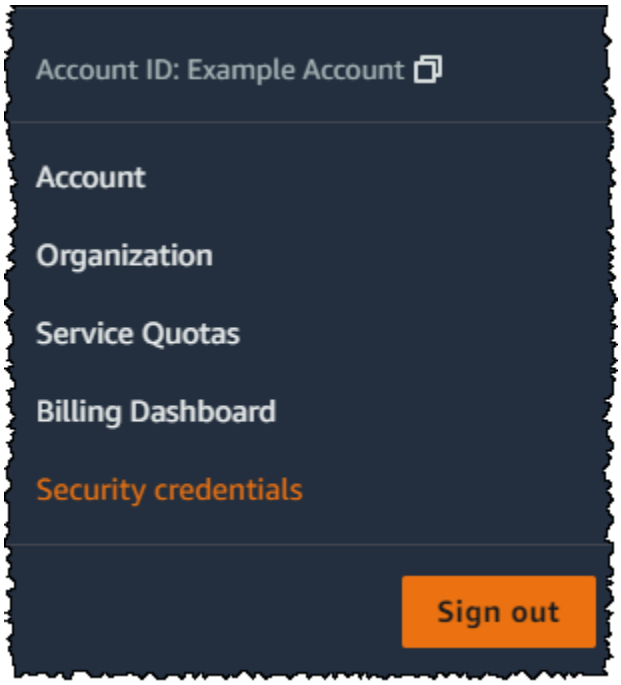
> **Note**
>
> Möglicherweise wird Ihnen unterschiedlicher Text angezeigt, z. B. Sign in using MFA (Melden Sie sich mit MFA an) und Troubleshoot your authentication device (Fehlerbehebung bei Ihrem Authentifizierungsgerät). Es stehen jedoch die gleichen Features zur Verfügung. In jedem Fall sollten Sie sich an AWS Support wenden, um Ihre MFA-Einstellung zu deaktivieren, wenn Sie die E-Mail-Adresse und Telefonnummer Ihres Kontos nicht mit anderen Authentifizierungsfaktoren verifizieren können.

So aktivieren Sie ein MFA-Gerät für Ihren Stammbenutzer (Konsole)

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

> **Note**
>
> Als Root-Benutzer können Sie sich nicht auf der Seite Sign in as IAM user (Als IAM-Benutzer anmelden) anmelden. Wenn Ihnen die Seite Sign in as IAM user (Als IAM-Benutzer anmelden) angezeigt wird, wählen Sie die Option Sign in using root user email (Mit Root-Benutzer-E-Mail anmelden) unten auf der Seite. Hilfe bei der Anmeldung als Root-Benutzer finden Sie im Benutzerhandbuch unter AWS Management Console Als AWS-Anmeldung Root-Benutzer anmelden.

2. Klicken Sie rechts in der Navigationsleiste auf den Kontonamen und wählen Sie dann Security Credentials (Sicherheitsanmeldeinformationen). Sofern erforderlich, wählen Sie Continue to Security Credentials (Weiter zu Sicherheitsanmeldeinformationen).



3. Erweitern Sie den Bereich Multi-Factor Authentication (MFA).
4. Wählen Sie Assign MFA device (MFA-Gerät zuweisen).
5. Geben Sie im Assistenten einen Device name (Gerätenamen) ein, wählen Sie Hardware TOTP token (Physisches TOTP-Token) und dann Next (Weiter).
6. Geben Sie im Feld Serial number (Seriennummer) die auf der Rückseite des MFA-Geräts aufgeführte Seriennummer ein.

7. Geben Sie im Feld MFA code 1 (MFA-Code 1) die am MFA-Gerät angezeigte sechsstellige Nummer ein. Sie müssen eventuell die Taste auf der Vorderseite des Geräts drücken, um die Zahl anzuzeigen.

8. Warten Sie 30 Sekunden, während das Gerät den Code aktualisiert, und geben Sie dann die nächste sechsstellige Nummer in das Feld MFA code 2 (MFA-Code 2) ein. Sie müssen eventuell die Taste auf der Vorderseite des Geräts drücken, um die zweite Zahl anzuzeigen.
9. Wählen Sie Add MFA (MFA hinzufügen). Das MFA-Gerät ist jetzt mit dem AWS-Konto verknüpft.

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Authentifizierungscodes. Wenn Sie die Codes erzeugen und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit dem Benutzer verknüpft, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. In diesem Fall können Sie das Gerät neu synchronisieren.

Wenn Sie sich das nächste Mal mit Ihren Stammbenutzer-Anmeldeinformationen anmelden, müssen Sie einen am MFA-Gerät angezeigten Code eingeben.

Ändern Sie das Passwort für Root-Benutzer des AWS-Kontos

Sie können die E-Mail-Adresse und das Passwort auf der Seite Sicherheitsanmeldeinformationen oder auf der Seite Konto ändern. Sie können auch Passwort vergessen? wählen auf der AWS Anmeldeseite, um Ihr Passwort zurückzusetzen.

Um das Passwort des Root-Benutzers zu ändern, müssen Sie sich als Root-Benutzer des AWS-Kontos und nicht als IAM-Benutzer anmelden. Wie Sie ein vergessenes Stammbenutzerpasswort zurücksetzen können, erfahren Sie unter Zurücksetzen eines verlorenen oder vergessenen Stammbenutzer-Passworts.

Um Ihr Passwort so gut wie möglich zu schützen, folgen Sie diesen bewährten Methoden:

- Ändern Sie Ihr Passwort regelmäßig.

- Halten Sie Ihr Passwort geheim, da jeder, der Ihr Passwort kennt, auf Ihr Konto zugreifen kann.
- Verwenden Sie für ein anderes Passwort AWS als für andere Websites.
- Vermeiden Sie Passwörter, die einfach zu erraten sind. Dazu gehören Passwörter wie `secretpassword`, `amazon` oder `123456`. Vermeiden Sie außerdem Dinge wie Wörter aus dem Wörterbuch, Ihren Namen, Ihre E-Mail-Adresse oder andere persönliche Informationen, die jemand leicht erhalten kann.

AWS Management Console

So ändern Sie das Passwort für den Stammbenutzer

> **Mindestberechtigungen**
>
> Für die folgenden Schritte sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - Sie müssen sich als AWS-Konto Root-Benutzer anmelden, wofür keine zusätzlichen AWS Identity and Access Management (IAM-) Berechtigungen erforderlich sind. Sie können diese Schritte nicht als IAM-Benutzer oder -Rolle ausführen.

1. Verwenden Sie Ihre AWS-Konto E-Mail-Adresse und Ihr Passwort, um sich AWS Management Consoleals Ihr Root-Benutzer des AWS-Kontos Konto anzumelden.
2. Wählen Sie oben rechts in der Konsole Ihren Kontonamen oder Ihre Kontonummer und dann Account (Konto) aus.
3. Wählen Sie auf der Seite Konto, neben den Kontoeinstellungen, Bearbeiten aus. Aus Sicherheitsgründen werden Sie aufgefordert, sich erneut zu authentifizieren.

   > **Note**
   >
   > Wenn die Option Bearbeiten nicht angezeigt wird, sind Sie wahrscheinlich nicht als Root-Benutzer für Ihr Konto angemeldet. Sie können die Kontoeinstellungen nicht ändern, wenn Sie als IAM-Benutzer oder als IAM-Rolle angemeldet sind.

4. Wählen Sie auf der Seite Kontoeinstellungen aktualisieren unter Passwort die Option Bearbeiten aus.
5. Füllen Sie auf der Seite Passwort aktualisieren die Felder für Aktuelles Passwort, Neues Passwort und Neues Passwort bestätigen aus.

> **Important**
>
> Achten Sie darauf, ein sicheres Passwort zu wählen. Obwohl Sie eine Kontokennwortrichtlinie für IAM-Benutzer festlegen können, gilt diese Richtlinie nicht für den Root-Benutzer.

AWS setzt voraus, dass Ihr Passwort die folgenden Bedingungen erfüllt:

- Es muss mindestens 8 Zeichen und maximal 128 Zeichen lang sein.
- Es muss mindestens drei der folgenden Zeichentypen enthalten: Großbuchstaben, Kleinbuchstaben, Zahlen und ! @ # $ % ^ & * () <> [] {} | _+-= Symbole.
- Es darf nicht mit Ihrem AWS-Konto Namen oder Ihrer E-Mail-Adresse identisch sein.

> **Note**
>
> AWS führt Verbesserungen am Anmeldeprozess ein. Eine dieser Verbesserungen ist das Erzwingen einer sicheren Passwortrichtlinie für Ihr Konto. Wenn AWS Ihr Konto aktualisiert hat, müssen Sie die zuvor beschriebene Passwortrichtlinie einhalten. Falls Ihr Konto AWS noch nicht aktualisiert wurde, wird diese Richtlinie noch AWS nicht durchgesetzt. Für ein sichereres Passwort empfehlen wir Ihnen jedoch dringend, die Richtlinien zu befolgen.

6. Wählen Sie Änderungen speichern aus.

AWS CLI or AWS SDK

Diese Aufgabe wird im AWS CLI oder durch eine API-Operation von einem der AWS SDKs nicht unterstützt. Sie können diese Aufgabe nur mit dem AWS Management Console ausführen.

Zurücksetzen eines verlorenen oder vergessenen Stammbenutzer-Passworts

Als Sie Ihre zum ersten Mal erstellt haben AWS-Konto, haben Sie eine E-Mail-Adresse und ein Passwort angegeben. Dies sind Ihre Root-Benutzer des AWS-Kontos Anmeldeinformationen. Wenn

Sie Ihr Stammbenutzer-Passwort vergessen, können Sie dieses über die AWS Management Console zurücksetzen.

So setzen Sie ihr Stammbenutzer-Passwort zurück:

1. Verwenden Sie Ihre AWS-Konto E-Mail-Adresse, um sich AWS Management Consoleals Root-Benutzer anzumelden, und wählen Sie dann Weiter.

   > **Note**
   >
   > Wenn Sie bei der AWS Management Console mit Ihren IAM-Benutzer-Anmeldedaten angemeldet sind, müssen Sie sich abmelden, bevor Sie das Stammbenutzer-Passwort zurücksetzen können. Wenn Sie sich auf der Anmeldeseite von IAM befinden, wählen Sie neben der Schaltfläche unten auf der Seite Mit Anmeldeinformationen des Stammkontos anmelden. Geben Sie ggf. Ihre Konto-E-Mail-Adresse an und wählen Sie Next (Weiter), um auf die Seite Root user sign in (Stammbenutzeranmeldung) zuzugreifen.

2. Wählen Sie Forgot your password? (Passwort vergessen?).

   > **Note**
   >
   > Wenn Sie ein IAM-Benutzer sind, steht diese Option nicht zur Verfügung. Die Option Passwort vergessen? steht nur für das Root-Benutzerkonto zur Verfügung. IAM-Benutzer müssen ihren Administrator bitten, ein vergessenes Passwort zurückzusetzen. Weitere Informationen finden Sie unter Ich habe mein IAM-Benutzerkennwort für mein AWS Konto vergessen. Wenn Sie sich über den anmelden AWS-Zugangsportal, finden Sie weitere Informationen unter Ihr IAM Identity Center-Benutzerkennwort zurücksetzen.

3. Geben Sie die E-Mail-Adresse an, die dem Konto zugeordnet ist. Anschließend geben Sie den CAPTCHA-Text ein und klicken Sie auf Continue (Weiter).

4. Suchen Sie in der E-Mail, die mit Ihrer verknüpft ist, AWS-Konto nach einer Nachricht von Amazon Web Services. Die E-Mail stammt von einer Adresse, die auf `@verify.signin.aws` endet. Befolgen Sie die Anweisungen in der E-Mail-Nachricht. Wenn Sie die E-Mail nicht in Ihrem Posteingang finden, überprüfen Sie Ihren Spam-Ordner. Wenn Sie keinen Zugriff mehr auf die E-Mail haben, finden Sie im AWS-Anmeldung Benutzerhandbuch weitere Informationen unter Ich habe keinen Zugriff auf die E-Mail-Adresse für mein AWS Konto.

Erstellen von Zugriffsschlüsseln für den Stammbenutzer

> **Warning**
>
> Wir empfehlen nachdrücklich, keine Zugriffsschlüsselpaare für Ihren Root-Benutzer zu erstellen. Da nur für wenige Aufgaben der Root-Benutzer erforderlich ist und Sie diese Aufgaben normalerweise selten ausführen, empfehlen wir, sich bei dem anzumelden, AWS Management Console um die Root-Benutzeraufgaben ausführen zu können. Bevor Sie Zugriffsschlüssel erstellen, prüfen Sie die Alternativen zu Langzeit-Zugriffsschlüsseln.

Wir empfehlen dies zwar nicht, Sie können jedoch Zugriffsschlüssel für Ihren Root-Benutzer erstellen, sodass Sie Befehle in der AWS Command Line Interface (AWS CLI) ausführen oder API-Operationen von einem der AWS SDKs aus mit Root-Benutzeranmeldedaten verwenden können. Bei der Erstellung eines Zugriffsschlüssels erstellen Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel als Set. Während der Erstellung des Zugriffsschlüssels haben Sie die AWS Möglichkeit, den Teil des Zugriffsschlüssels mit dem geheimen Zugriffsschlüssel anzusehen und herunterzuladen. Wenn Sie ihn nicht herunterladen oder ihn verlieren, können Sie den Zugriffsschlüssel löschen und einen neuen erstellen. Sie können Root-Benutzerzugriffsschlüssel mit der Konsole oder der AWS API erstellen. AWS CLI

Ein neu erstellter Zugriffsschlüssel hat den Status aktiv, das heißt, Sie können den Zugriffsschlüssel für die CLI und API-Aufrufe verwenden. Sie können dem Root-Benutzer bis zu zwei Zugriffsschlüssel zuweisen.

Zugriffsschlüssel, die nicht verwendet werden, sollten deaktiviert werden. Sobald ein Zugriffsschlüssel inaktiv ist, können Sie ihn nicht für API-Aufrufe verwenden. Inaktive Schlüssel werden immer noch auf Ihr Limit angerechnet. Sie können einen Zugriffsschlüssel jederzeit erstellen oder löschen. Wenn Sie jedoch einen Zugriffsschlüssel löschen, wird er endgültig entfernt und kann nicht mehr abgerufen werden.

AWS Management Console

Um einen Zugriffsschlüssel für das zu erstellen Root-Benutzer des AWS-Kontos

> **Mindestberechtigungen**
>
> Für die folgenden Schritte sind mindestens folgende IAM-Berechtigungen erforderlich:

- Sie müssen sich als AWS-Konto Root-Benutzer anmelden, wofür keine zusätzlichen AWS Identity and Access Management (IAM-) Berechtigungen erforderlich sind. Sie können diese Schritte nicht als IAM-Benutzer oder -Rolle ausführen.

1. Verwenden Sie Ihre AWS-Konto E-Mail-Adresse und Ihr Passwort, um sich bei Getting Started mit dem AWS Management Console Namen Ihr Root-Benutzer des AWS-Kontos anzumelden.
2. Wählen Sie oben rechts in der Konsole Ihren Kontonamen oder Ihre Kontonummer und dann Sicherheitsanmeldeinformationen.
3. Wählen Sie im Abschnitt Access keys (Zugriffsschlüssel) Create access key (Zugriffsschlüssel erstellen). Wenn diese Option nicht verfügbar ist, verfügen Sie bereits über die maximale Anzahl an Zugriffsschlüsseln. Sie müssen einen der vorhandenen Zugriffsschlüssel löschen, bevor Sie einen neuen Schlüssel erstellen können. Weitere Informationen finden Sie unter IAM-Objekt-Kontingente.
4. Lesen Sie auf der Seite Alternativen zu Root-Benutzer-Zugriffsschlüsseln die Sicherheitsempfehlungen. Um fortzufahren, aktivieren Sie das Kontrollkästchen und wählen Sie dann Zugriffsschlüssel erstellen aus.
5. Auf der Seite Zugriffsschlüssel abrufen wird Ihre Zugriffsschlüssel-ID angezeigt.
6. Wählen Sie unter Geheimer Zugriffsschlüssel die Option Anzeigen aus, kopieren Sie dann die Zugriffsschlüssel-ID und den geheimen Schlüssel aus Ihrem Browserfenster und fügen Sie sie an einem sicheren Ort ein. Alternativ können Sie CSV-Datei herunterladen wählen, um eine Datei mit dem Namen `rootkey.csv` herunterzuladen, die die Zugriffsschlüssel-ID und den geheimen Schlüssel enthält. Speichern Sie die Datei an einem sicheren Ort.
7. Wählen Sie Erledigt aus. Wenn Sie den Zugriffsschlüssel nicht mehr benötigen, empfehlen wir Ihnen, ihn zu löschen oder ihn zumindest zu deaktivieren, damit ihn niemand missbrauchen kann.

AWS CLI & SDKs

So erstellen Sie einen Zugriffsschlüssel für den Root-Benutzer

> **Note**
>
> Um den folgenden Befehl oder die folgende API-Operation als Root-Benutzer auszuführen, müssen Sie bereits über ein aktives Zugriffsschlüsselpaar verfügen. Wenn Sie über keine Zugriffsschlüssel verfügen, erstellen Sie den ersten Zugriffsschlüssel mithilfe der AWS Management Console. Anschließend können Sie die Anmeldeinformationen dieses ersten Zugriffsschlüssels zusammen mit dem verwenden, AWS CLI um den zweiten Zugriffsschlüssel zu erstellen oder einen Zugriffsschlüssel zu löschen.

- AWS CLI: Was ich bin create-access-key

  Example

  ```
  $ aws iam create-access-key
  {
      "AccessKey": {
          "UserName": "MyUserName",
          "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
          "Status": "Active",
          "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
          "CreateDate": "2021-04-08T19:30:16+00:00"
      }
  }
  ```

- AWS API: CreateAccessKeyin der IAM-API-Referenz.

Löschen von Zugriffsschlüsseln für den Stammbenutzer

Sie können die AWS Management Console, die AWS CLI oder die AWS API verwenden, um die Root-Benutzerzugriffsschlüssel zu löschen.

AWS Management Console

So löschen Sie einen Zugriffsschlüssel für den Root-Benutzer

> **Mindestberechtigungen**
>
> Für die folgenden Schritte sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - Sie müssen sich als AWS-Konto Root-Benutzer anmelden, wofür keine zusätzlichen AWS Identity and Access Management (IAM-) Berechtigungen erforderlich sind. Sie können diese Schritte nicht als IAM-Benutzer oder -Rolle ausführen.

1. Verwenden Sie Ihre AWS-Konto E-Mail-Adresse und Ihr Passwort, um sich bei Getting Started mit dem AWS Management Console Namen Ihr Root-Benutzer des AWS-Kontos anzumelden.
2. Wählen Sie oben rechts in der Konsole Ihren Kontonamen oder Ihre Kontonummer und dann Sicherheitsanmeldeinformationen.
3. Wählen Sie im Abschnitt Zugriffsschlüssel den Zugriffsschlüssel aus, den Sie löschen möchten, und wählen Sie dann unter Aktionen die Option Löschen aus.

   > **Note**
   >
   > Alternativ können Sie einen Zugriffsschlüssel deaktivieren, anstatt ihn dauerhaft zu löschen. Auf diese Weise können Sie ihn in Zukunft wieder verwenden, ohne die Schlüssel-ID oder den geheimen Schlüssel ändern zu müssen. Solange der Schlüssel inaktiv ist, schlagen alle Versuche, ihn in Anfragen an die AWS API zu verwenden, fehl und es wird die Fehlermeldung „Zugriff verweigert“ angezeigt.

4. Wählen Sie im Dialogfeld <acess key ID> löschen die Option Deaktivieren aus, geben Sie die Zugriffsschlüssel-ID ein, um zu bestätigen, dass Sie sie löschen möchten, und wählen Sie dann Löschen.

AWS CLI & SDKs

So löschen Sie einen Zugriffsschlüssel für den Root-Benutzer

> **Mindestberechtigungen**
>
> Für die folgenden Schritte sind mindestens folgende IAM-Berechtigungen erforderlich:
>
> - Sie müssen sich als AWS-Konto Root-Benutzer anmelden, wofür keine zusätzlichen AWS Identity and Access Management (IAM-) Berechtigungen erforderlich sind. Sie können diese Schritte nicht als IAM-Benutzer oder -Rolle ausführen.

- AWS CLI: war ich delete-access-key

  Example

  ```
  $ aws iam delete-access-key \
      --access-key-id AKIAIOSFODNN7EXAMPLE
  ```

  Dieser Befehl erzeugt keine Ausgabe, wenn er erfolgreich ist.
- AWS API: DeleteAccessKey

Aufgaben, die Stammbenutzer-Anmeldeinformationen erfordern

> **Important**
>
> Haben Sie Probleme bei der Anmeldung AWS? Stellen Sie sicher, dass Sie sich auf der richtigen AWS -Anmeldeseite für Ihren Benutzertyp befinden. Wenn Sie der Root-Benutzer des AWS-Kontos (Kontoinhaber) sind, können Sie sich AWS mit den Anmeldeinformationen anmelden, die Sie bei der Erstellung des eingerichtet haben AWS-Konto. Wenn Sie ein IAM-Benutzer sind, kann Ihr Kontoadministrator Ihnen die Anmeldeinformationen bereitstellen, mit denen Sie sich bei AWS anmelden können. Wenn Sie Support anfordern müssen, verwenden Sie nicht den Feedback-Link auf dieser Seite, da das Formular beim AWS Dokumentationsteam eingegangen ist, nicht AWS Support. Wählen Sie stattdessen auf der Seite „Kontaktieren Sie uns“ die Option Immer noch nicht in Ihr AWS Konto einloggen und wählen Sie dann eine der verfügbaren Support-Optionen aus.

Wir empfehlen Ihnen, einen Administratorbenutzer für die Ausführung täglicher Aufgaben und AWS IAM Identity Center den Zugriff auf AWS Ressourcen zu konfigurieren. Sie können die unten aufgeführten Aufgaben aber nur ausführen, wenn Sie als Stammbenutzer eines Kontos angemeldet sind.

Aufgaben der Kontoverwaltung

- Ändern Ihrer Kontoeinstellungen. Dazu gehören der Kontoname, die E-Mail-Adresse, das Passwort des Stammbenutzers und die Zugriffsschlüssel des Stammbenutzers. Für andere Kontoeinstellungen, wie Kontaktinformationen, bevorzugte Zahlungswährung und AWS-Regionen, sind keine Root-Benutzeranmeldedaten erforderlich.
- Stellen Sie IAM-Benutzerberechtigungen wieder her. Wenn der einzige IAM-Administrator versehentlich die eigenen Berechtigungen widerruft, können Sie sich als Stammbenutzer anmelden, um Richtlinien zu bearbeiten und diese Berechtigungen wiederherzustellen.
- Schließe deine AWS-Konto.

  Weitere Informationen finden Sie unter den folgenden Themen:

  - Wie übertrage ich meine Inhaberschaft AWS-Konto an eine andere Entität? .
  - Wie schließe ich meine AWS-Konto? .
  - Schließt einen eigenständigen Computer AWS-Konto.

Aufgaben zur Abrechnung

- Aktivieren Sie IAM-Zugriff auf die Konsole für Fakturierung und Kostenmanagement.
- Einige Abrechnungsaufgaben sind auf den Root-Benutzer beschränkt. Weitere Informationen finden Sie AWS-Konto im AWS Billing Benutzerhandbuch unter Verwaltung eines.
- Anzeigen bestimmter Steuerrechnungen. Ein IAM-Benutzer mit der ViewBilling Berechtigung aws-portal: kann Mehrwertsteuerrechnungen aus AWS Europa einsehen und herunterladen, nicht jedoch Rechnungen von AWS Inc. oder Amazon Internet Services Private Limited (AISPL).

AWS GovCloud (US) Aufgaben

- Registrieren Sie sich für AWS GovCloud (US).
- Fordern Sie die Zugangsschlüssel für den Root-Benutzer für das AWS GovCloud (US) Konto an von AWS Support.

Amazon EC2 EC2-Aufgabe

- Sie haben sich im Reserved Instance Marketplace als Verkäufer registriert.

AWS KMS Aufgabe

- Falls ein AWS Key Management Service Schlüssel nicht mehr verwaltet werden kann, kann er von einem Administrator wiederhergestellt werden, indem er sich an die primäre Telefonnummer Ihres Root-Benutzers AWS Support wendet, um die Autorisierung einzuholen, indem er das Einmalpasswort des Tickets bestätigt. AWS Support

Aufgabe von Amazon Mechanical Turk

- Verknüpfen Sie Ihr Konto AWS-Konto mit Ihrem MTurk Requester-Konto.

Aufgaben von Amazon Simple Storage Service

- Konfigurieren Sie einen Amazon-S3-Bucket, um MFA (Multi-Faktor-Authentifizierung) zu aktivieren.
- Bearbeiten oder löschen Sie eine Amazon S3 S3-Bucket-Richtlinie, die alle Principals verweigert.

Amazon Simple Queue Service-Aufgabe

- Bearbeiten oder löschen Sie eine Amazon SQS SQS-Ressourcenrichtlinie, die alle Principals verweigert.

Fehlerbehebung bei Problemen mit dem Root-Benutzer

Verwenden Sie die hier aufgeführten Informationen, um Probleme im Zusammenhang mit dem Root-Benutzer eines AWS-Konto zu beheben.

Ich kann keine Aufgaben ausführen, die ich erwarte, wenn ich als Root-Benutzer des Kontos angemeldet bin

Wenn Sie keine Aufgaben erledigen können, wenn Sie als Root-Benutzer des Kontos angemeldet sind, ist Ihr Konto möglicherweise Mitglied einer Organisation in AWS Organizations. Wenn dies der Fall ist und Ihr Organisationsadministrator eine Service-Kontrollrichtlinie (SCP) verwendet, um die Berechtigungen Ihres Kontos einzuschränken, sind alle Benutzer, einschließlich des Root-Benutzers,

betroffen. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien im AWS Organizations - Benutzerhandbuch.

Ich habe das Root-Benutzer-Passwort für mein AWS-Konto vergessen

Wenn Sie ein Root-Benutzer sind und Ihr Passwort verloren oder vergessen haben AWS-Konto, können Sie Ihr Passwort zurücksetzen. Sie müssen die E-Mail-Adresse kennen, mit der das AWS-Konto erstellt wurde, und Sie müssen Zugriff auf das E-Mail-Konto haben. Weitere Informationen finden Sie unter Zurücksetzen eines verlorenen oder vergessenen Stammbenutzer-Passworts.

Ich habe keinen Zugriff auf die E-Mail für meine AWS-Konto

Wenn Sie eine erstellen AWS-Konto, geben Sie eine E-Mail-Adresse und ein Passwort an. Dies sind die Anmeldeinformationen für den Root-Benutzer des AWS-Kontos. Wenn Sie sich nicht sicher sind, welche E-Mail-Adresse mit Ihrer verknüpft ist AWS-Konto, suchen Sie nach Nachrichten, die von `@signin.aws` oder `@verify.signin.aws` an eine E-Mail-Adresse Ihrer Organisation gesendet wurden, die möglicherweise zum Öffnen von verwendet wurde AWS-Konto.

Wenn Sie die E-Mail-Adresse kennen, aber keinen Zugriff mehr auf die E-Mail haben, versuchen Sie zunächst, den Zugriff auf die E-Mail mit einer der folgenden Optionen wiederherzustellen:

- Wenn Sie Eigentümer der Domain für die E-Mail-Adresse sind, können Sie eine gelöschte E-Mail-Adresse wiederherstellen. Alternativ können Sie ein Catch-All für Ihr E-Mail-Konto einrichten, das alle Nachrichten abfängt, die an E-Mail-Adressen gesendet werden, die auf dem Mail-Server nicht mehr vorhanden sind, und sie an eine andere E-Mail-Adresse umleitet.
- Wenn die E-Mail-Adresse des Kontos Teil Ihres Unternehmens-E-Mail-Systems ist, empfehlen wir Ihnen, sich mit Ihren IT-Systemadministratoren in Verbindung zu setzen. Sie können Ihnen vielleicht helfen, den Zugriff auf die E-Mail wiederherzustellen.

Falls Sie sich immer noch nicht bei Ihrem anmelden können AWS-Konto, finden Sie alternative Support-Optionen unter Kontaktieren Sie uns.

Ähnliche Informationen

Die folgenden Artikel enthalten zusätzliche Informationen zum Arbeiten mit dem Root-Benutzer.

- Was sind einige bewährte Methoden zur Sicherung meiner AWS-Konto und ihrer Ressourcen?
- Wie kann ich eine EventBridge Ereignisregel erstellen, die mich darüber informiert, dass mein Root-Benutzer verwendet wurde?

- Überwachen Sie Root-Benutzer des AWS-Kontos Aktivitäten und benachrichtigen Sie sie
- IAM-Root-Benutzeraktivitäten überwachen

IAM-Benutzer

> **Important**
>
> Bewährte IAM-Methoden empfehlen, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um mit temporären Anmeldeinformationen zuzugreifen AWS , anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen zu verwenden.

Ein AWS Identity and Access Management (IAM-) Benutzer ist eine Entität, in der Sie erstellen. AWS Der IAM-Benutzer steht für den menschlichen Benutzer oder die Arbeitslast, mit dem der IAM-Benutzer interagiert. AWS Ein Benutzer AWS besteht aus einem Namen und Anmeldeinformationen.

Ein IAM-Benutzer mit Administratorberechtigungen ist nicht dasselbe wie der Root-Benutzer des AWS-Kontos. Weitere Informationen zum Stammbenutzer finden Sie unter Root-Benutzer des AWS-Kontos.

Wie AWS identifiziert man einen IAM-Benutzer

Bei der Erstellung eines IAM-Benutzers erstellt IAM die folgenden Elemente zum Identifizieren dieses Benutzers:

- Einen „Anzeigename“ für den IAM-Benutzer. Dies ist der Name, den Sie beim Erstellen des IAM-Benutzers angegeben haben, z. B. `Richard` oder `Anaya`. Diese Namen werden in der AWS Management Console angezeigt.
- Ein Amazon-Ressourcenname (ARN) für den IAM-Benutzer. Sie verwenden den ARN, wenn Sie den IAM-Benutzer in allen Bereichen eindeutig identifizieren müssen. AWS Sie können z. B. einen ARN verwenden, um den IAM-Benutzer als `Principal` in einer IAM-Richtlinie für einen Amazon-S3-Bucket festzulegen. Ein ARN für einen IAM-Benutzer könnte folgendes Format aufweisen:

  `arn:aws:iam::`*`account-ID-without-hyphens`*`:user/Richard`
- Eine eindeutige ID für den IAM-Benutzer. Diese ID wird nur zurückgegeben, wenn Sie die API oder Tools für Windows PowerShell verwenden oder AWS CLI um den IAM-Benutzer zu erstellen. Sie sehen diese ID nicht in der Konsole.

Weitere Informationen zu diesen IDs finden Sie unter IAM-IDs.

IAM-Benutzer und Anmeldeinformationen

Sie können je nach AWS den IAM-Benutzeranmeldedaten auf unterschiedliche Weise darauf zugreifen:

- Konsolenpasswort: Ein Passwort, das der IAM-Benutzer eingeben kann, um sich bei interaktiven Sitzungen wie der AWS Management Console anzumelden. Wenn Sie das Passwort (Konsolenzugriff) für einen IAM-Benutzer deaktivieren, kann er sich nicht AWS Management Console mit seinen Anmeldeinformationen anmelden. Es ändert weder ihre Berechtigungen noch verhindert, dass sie mit einer angenommenen Rolle auf die Konsole zugreifen.
- Zugriffsschlüssel: Werden verwendet, um programmatische Aufrufe an AWS zu tätigen. Es gibt jedoch sicherere Alternativen, die Sie in Betracht ziehen sollten, bevor Sie Zugriffsschlüssel für IAM-Benutzer erstellen. Weitere Informationen finden Sie unter Überlegungen und Alternativen für Schlüssel für den langfristigen Zugriff in der Allgemeine AWS-Referenz. Wenn der IAM-Benutzer über aktive Zugriffsschlüssel verfügt, funktionieren diese weiterhin und ermöglichen den AWS CLI Zugriff über Tools für Windows PowerShell, AWS API oder die AWS Console Mobile Application.
- SSH-Schlüssel zur Verwendung mit CodeCommit: Ein öffentlicher SSH-Schlüssel im OpenSSH-Format, mit dem sich authentifizieren kann. CodeCommit
- Serverzertifikate: SSL/TLS-Zertifikate, mit denen Sie sich bei einigen Diensten authentifizieren können. AWS Wir empfehlen Ihnen, AWS Certificate Manager (ACM) für die Bereitstellung, Verwaltung und Bereitstellung Ihrer Serverzertifikate zu verwenden. Verwenden Sie IAM nur für die Unterstützung von HTTPS-Verbindungen in einer Region, die nicht von ACM unterstützt wird. Informationen darüber, welche Regionen ACM unterstützen, finden Sie unter AWS Certificate Manager -Endpunkte und -Kontingente in der Allgemeine AWS-Referenz.

Sie können die richtigen Anmeldeinformationen für Ihren IAM-Benutzer wählen. Wenn Sie die AWS Management Console verwenden, um einen IAM-Benutzer zu erstellen, müssen Sie mindestens ein Konsolenpasswort oder Zugriffsschlüssel eingeben. Standardmäßig hat ein brandneuer IAM-Benutzer, der mit der AWS CLI oder AWS API erstellt wurde, keinerlei Anmeldeinformationen. Sie müssen die Art der Anmeldeinformationen für einen IAM-Benutzer je nach dem Anwendungsfall erstellen.

Sie haben die folgenden Optionen zum Verwalten von Passwörtern, Zugriffsschlüsseln und Multi-Faktor-Authentifizierung (MFA)-Geräten:

- Verwalten von Passwörtern für Ihre IAM-Benutzer. Erstellen und ändern Sie die Passwörter, die Zugriff auf die AWS Management Console ermöglichen. Legen Sie eine Passwortrichtlinie fest, um eine Mindest-Passwortkomplexität zu erzwingen. Ermöglichen Sie Benutzern, ihre eigenen Passwörter zu ändern.
- Verwalten der Zugriffsschlüssel für Ihre IAM-Benutzer. Erstellen und aktualisieren Sie Zugriffsschlüssel für den programmgesteuerten Zugriff auf die Ressourcen in Ihrem Konto.
- Aktivieren Sie die Multi-Faktor-Authentifizierung (MFA) für den IAM-Benutzer. Als bewährte Methode empfehlen wir, dass Sie Multi-Faktor-Authentifizierung (MFA) für alle IAM-Benutzer in Ihrem Konto benötigen. Bei MFA müssen Anwender zwei Formen der Identifikation bereitstellen: Zuerst geben sie die Anmeldedaten an, die Teil ihrer Benutzeridentität sind (ein Passwort oder Zugriffsschlüssel). Darüber hinaus bieten sie einen temporären Zahlencode, der auf einem Hardwaregerät oder durch eine Anwendung auf einem Smartphone oder Tablet generiert wird.
- Suchen von ungenutzten Passwörtern und Zugriffsschlüsseln. Jeder, der ein Passwort oder Zugangsschlüssel für Ihr Konto oder einen IAM-Benutzer in Ihrem Konto hat, hat Zugriff auf Ihre AWS Ressourcen. Als bewährte Sicherheitsmethode empfiehlt es sich, Passwörter und Zugriffsschlüssel zu entfernen, wenn die Benutzer diese nicht mehr benötigen.
- Herunterladen eines Berichts zu Anmeldeinformationen für Ihr Konto. Sie können einen Bericht zu Anmeldeinformationen erstellen und herunterladen. In diesem Bericht sind alle IAM-Benutzer unter Ihrem Konto mit dem Status ihrer verschiedenen Anmeldeinformationen aufgeführt (z. B. Passwörter, Zugriffsschlüssel und MFA-Geräte). Für Passwörter und Zugriffsschlüssel zeigt der Bericht an, wann das Passwort oder der Zugriffsschlüssel zuletzt verwendet wurde.

IAM-Benutzer und Berechtigungen

Standardmäßig besitzt ein neuer IAM-Benutzer überhaupt keine Berechtigungen. Sie sind nicht berechtigt, AWS Operationen durchzuführen oder auf AWS Ressourcen zuzugreifen. Ein Vorteil, über einzelne IAM-Benutzer zu verfügen, besteht darin, dass Sie jedem Benutzer individuell Berechtigungen zuweisen können. Möglicherweise weisen Sie einigen Benutzern Administratorberechtigungen zu, die dann Ihre AWS Ressourcen verwalten und sogar andere IAM-Benutzer erstellen und verwalten können. In den meisten Fällen möchten Sie die Berechtigungen eines Benutzers jedoch auf die Aufgaben (AWS Aktionen oder Operationen) und Ressourcen beschränken, die für den Job benötigt werden.

Angenommen, wir haben einen Benutzer namens Diego. Wenn Sie IAM-Benutzer `Diego` erstellen, weisen Sie ihm auch Berechtigungen zu, die es ihm ermöglichen, eine bestimmte Amazon-EC2-Instance zu starten und Informationen aus einer Tabelle in einer Amazon-RDS-Datenbank

zu lesen (GET). Weitere Informationen dazu, wie Sie Benutzer erstellen und ihnen erstmalige Anmeldeinformationen und Berechtigungen gewähren, finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto. Anweisungen zum Ändern der Berechtigungen für vorhandene Benutzer finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer. Anweisungen zum Ändern des Passworts oder der Zugriffsschlüssel des Benutzers finden Sie unter Benutzerpasswörter verwalten in AWS und Verwalten der Zugriffsschlüssel für IAM-Benutzer.

Sie können Ihren IAM-Benutzern auch eine Berechtigungsgrenze hinzufügen. Eine Berechtigungsgrenze ist eine erweiterte Funktion, mit der Sie mithilfe AWS verwalteter Richtlinien die maximalen Berechtigungen einschränken können, die eine identitätsbasierte Richtlinie einem IAM-Benutzer oder einer IAM-Rolle gewähren kann. Weitere Informationen zu diesen Richtlinienarten und ihrer Verwendung finden Sie unter Berechtigungen und Richtlinien in IAM.

IAM-Benutzer und Konten

Jeder IAM-Benutzer kann nur einem einzigen AWS-Konto zugeordnet sein. Da IAM-Benutzer in Ihrem System definiert sind AWS-Konto, müssen Sie für sie keine Zahlungsmethode hinterlegt haben. AWS Jede AWS Aktivität, die von IAM-Benutzern in Ihrem Konto ausgeführt wird, wird Ihrem Konto in Rechnung gestellt.

Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

IAM-Benutzer als Servicekonten

Ein IAM-Benutzer ist eine Ressource in IAM, der Anmeldeinformationen und Berechtigungen zugeordnet sind. Ein IAM-Benutzer kann eine Person oder eine Anwendung darstellen, die ihre Anmeldeinformationen für AWS -Anforderungen verwendet. Dies wird in der Regel als Servicekonto bezeichnet. Falls Sie die langfristigen Anmeldeinformationen eines IAM-Benutzers in Ihrer Anwendung verwenden möchten, betten Sie keine Zugriffsschlüssel direkt in den Anwendungscode ein. Die AWS SDKs und die AWS Command Line Interface ermöglichen es Ihnen, Zugriffsschlüssel an bekannten Orten zu platzieren, sodass Sie sie nicht im Code speichern müssen. Weitere Informationen finden Sie unter Manage IAM User Access Keys Properly (Ordnungsgemäßes Verwalten von IAM-Benutzerzugriffsschlüsseln) in der Allgemeine AWS-Referenz. Alternativ und als bewährte Methode können Sie temporäre Sicherheitsanmeldeinformationen (IAM-Rollen) anstelle langfristiger Zugriffsschlüssel verwenden.

Erstellen eines IAM-Benutzers in Ihrem AWS-Konto

Follow us on Twitter

> **Important**
>
> Bewährte IAM-Methoden empfehlen, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um mit temporären Anmeldeinformationen zuzugreifen AWS , anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen zu verwenden.

> **Note**
>
> Wenn Sie diese Seite gefunden haben, weil Sie nach Informationen über die Produktwerbe-API für den Verkauf von Amazon-Produkten auf Ihrer Website suchen, schlagen Sie in der Product Advertising API 5.0 Dokumentation nach.
> Wenn Sie von der IAM-Konsole zu dieser Seite weitergeleitet wurden, ist es möglich, dass in Ihrem Konto kein IAM-Benutzer vorhanden ist, auch wenn Sie angemeldet sind. Sie sind möglicherweise als Root-Benutzer des AWS-Kontos oder mit temporären Anmeldeinformationen angemeldet. Weitere Informationen zu diesen IAM-Rollen finden Sie unter IAM-Identitäten (Benutzer, Gruppen und Rollen).

Der Prozess, einen Benutzer anzulegen und ihm die Durchführung von Arbeitsaufgaben zu ermöglichen, besteht aus den folgenden Schritten:

1. Erstellen Sie den Benutzer in den AWS Management Console AWS CLI Tools für Windows PowerShell oder mithilfe einer AWS API-Operation. Wenn Sie den Benutzer in der erstellen AWS Management Console, werden die Schritte 1—4 automatisch ausgeführt, je nachdem, was Sie ausgewählt haben. Wenn Sie Benutzer programmgesteuert erstellen, müssen Sie diese Schritte einzeln ausführen.
2. Erstellen Sie Anmeldeinformationen für den Benutzer, je nach Art des für den Benutzer erforderlichen Zugriffs:
   - Konsolenzugriff aktivieren — optional: Wenn der Benutzer auf die zugreifen muss AWS Management Console, erstellen Sie ein Passwort für den Benutzer. Die Deaktivierung des Konsolenzugriffs für einen Benutzer verhindert, dass er sich mit seinem Benutzernamen und Passwort bei der AWS Management Console anmeldet. Es ändert weder ihre Berechtigungen noch verhindert, dass sie mit einer angenommenen Rolle auf die Konsole zugreifen.

> **Tip**
>
> Erstellen Sie nur die Anmeldeinformationen, die der Benutzer braucht. Erstellen Sie beispielsweise für einen Benutzer, der Zugriff nur über die benötigt AWS Management Console, keine Zugriffsschlüssel.

3. Erteilen Sie dem Benutzer die zum Ausführen der erforderlichen Aufgaben notwendigen Berechtigungen, indem Sie den Benutzer zu einer oder mehreren Gruppen hinzufügen. Sie können auch Berechtigungen erteilen, indem Sie Ihre Berechtigungsrichtlinien direkt dem Benutzer hinzufügen. Allerdings empfehlen wir stattdessen, dass Sie Ihre Benutzer Gruppen zuordnen und die Berechtigungen anhand von zu diesen Gruppen angefügten Richtlinien verwalten. Sie können auch eine Berechtigungsgrenze verwenden, um die Berechtigungen von Benutzern zu begrenzen, obwohl dies nicht üblich ist.
4. (Optional) Fügen Sie dem Benutzer Metadaten durch Anfügen von Tags hinzu. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
5. Teilen Sie dem Benutzer die erforderlichen Anmeldeinformationen mit. Dies umfasst das Passwort und die Konsolen-URL für die Anmelde-Website des Kontos, auf der der Benutzer diese Anmeldeinformationen eingeben muss. Weitere Informationen finden Sie unter So melden sich IAM-Benutzer an AWS.
6. (Optional) Konfigurieren Sie die Multifaktor-Authentifizierung (MFA) für den Benutzer. Bei MFA muss der Benutzer bei jeder Anmeldung einen one-time-use Code angeben. AWS Management Console
7. (Optional) Geben Sie den Benutzern Berechtigungen zum Verwalten ihrer eigenen Sicherheitsanmeldeinformationen. (Standardmäßig verfügen die Benutzer nicht über Berechtigungen zum Verwalten ihrer eigenen Sicherheitsanmeldeinformationen.) Weitere Informationen finden Sie unter Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können.

Weitere Informationen zu den Berechtigungen, die Sie zum Erstellen eines Benutzers benötigen, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

Themen

- Erstellen von IAM-Benutzern (Konsole)
- Erstellen von IAM-Benutzern (AWS CLI)
- IAM-Benutzer erstellen (API)AWS

Erstellen von IAM-Benutzern (Konsole)

Sie können den verwenden AWS Management Console , um IAM-Benutzer zu erstellen.

So erstellen Sie einen IAM-Benutzer (Konsole)

1. Folgen Sie dem Anmeldeverfahren, das Ihrem Benutzertyp entspricht, wie im Abschnitt So melden Sie sich bei AWS an im AWS -Anmelde-Benutzerhandbuch beschrieben.
2. Wählen Sie auf der Console Home (Konsolen-Startseite) den IAM-Service aus.
3. Wählen Sie im Navigationsbereich Users (Benutzer) und dann Add users (Benutzer hinzufügen) aus.
4. Geben Sie auf der Seite Specify user details (Benutzerdetails angeben) unter User details (Benutzerdetails) in das Feld User name (Benutzername) den Namen für den neuen Benutzer ein. Dies ist der Anmeldename für AWS.

   > **Note**
   >
   > Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS. Benutzernamen können eine Kombination aus bis zu 64 Buchstaben, Ziffern und den folgenden Zeichen enthalten: Plus (+), Gleichheitszeichen (=), Komma (,), Punkt (.), at-Zeichen (@), Unterstrich (_) und Bindestrich (-). Namen müssen innerhalb eines Kontos eindeutig sein. Es wird hierbei nicht zwischen Groß- und Kleinschreibung unterschieden. So können Sie beispielsweise keine zwei Gruppen mit Namen TESTBENUTZER und testbenutzer erstellen. Wenn ein Benutzername in einer Richtlinie oder als Teil eines ARN verwendet wird, ist die Groß-/Kleinschreibung des Namens zu beachten. Wenn Kunden in der Konsole ein Benutzername angezeigt wird, beispielsweise während des Anmeldevorgangs, wird die Groß-/Kleinschreibung des Benutzernamens nicht beachtet.

5. Wählen Sie Benutzerzugriff auf bereitstellen aus — AWS Management Console optional. Dadurch werden AWS Management Console Anmeldeinformationen für den neuen Benutzer generiert.

   Sie werden gefragt, ob Sie einer Person Zugriff auf die Konsole gewähren. Wir empfehlen, dass Sie Benutzer in IAM Identity Center und nicht in IAM erstellen.

   - Um zur Erstellung des Benutzers in IAM Identity Center zu wechseln, wählen Sie Specify a user in Identity Center (Benutzer in Identity Center angeben) aus.

Wenn Sie IAM Identity Center nicht aktiviert haben, gelangen Sie durch Auswahl dieser Option zur Serviceseite in der Konsole, auf der Sie den Service aktivieren können. Einzelheiten zu diesem Verfahren finden Sie im Benutzerhandbuch unter Erste Schritte mit den häufigsten Aufgaben in IAM Identity CenterAWS IAM Identity Center

Wenn Sie IAM Identity Center aktiviert haben, gelangen Sie durch Auswahl dieser Option zur Seite Specify user details (Benutzerdetails angeben) in IAM Identity Center. Einzelheiten zu diesem Verfahren finden Sie im Benutzerhandbuch unter AWS IAM Identity Center Benutzer hinzufügen

- Wenn Sie IAM Identity Center nicht verwenden können, wählen Sie I want to create an IAM user (Ich möchte einen IAM-Benutzer erstellen) aus und fahren Sie mit diesem Verfahren fort.

a. Wählen Sie für Console password (Konsolenpasswort) eine der nachstehenden Optionen aus:

   - Autogenerated password (Automatisch generiertes Passwort) – Jeder Benutzer erhält ein zufallsgeneriertes Passwort, das die Kontopasswortrichtlinie erfüllt. Sie können das Passwort auf der Seite Retrieve passwort (Passwort abrufen) ansehen oder herunterladen.
   - Custom password (Benutzerdefiniertes Passwort) – Jedem Benutzer wird das von Ihnen in das Feld eingegebene Passwort zugewiesen.

b. (Optional) Users must create a new password at next sign-in (recommended) (Benutzer müssen bei der nächsten Anmeldung ein neues Passwort erstellen (empfohlen)) ist standardmäßig ausgewählt, um sicherzustellen, dass der Benutzer gezwungen ist, sein Passwort bei der ersten Anmeldung zu ändern.

> **Note**
>
> Wenn ein Administrator die Kontopasswortrichtlinie Allow users to change their own password (Benutzer dürfen ihr eigenes Kennwort ändern) aktiviert hat, bewirkt dieses Kontrollkästchen nichts. Andernfalls wird automatisch eine verwaltete AWS -Richtlinie mit dem Namen `IAMUserChangePassword` an die neuen Benutzer angehängt. Die Richtlinie gewährt ihnen die Erlaubnis, ihre eigenen Passwörter zu ändern.

6. Klicken Sie auf Weiter.

7. Geben Sie auf der Seite Set permissions (Berechtigungen festlegen) an, wie Sie die Berechtigungen für diesen Benutzer zuweisen möchten. Wählen Sie eine der folgenden drei Optionen aus:

   - Add user to group (Benutzer zur Gruppe hinzufügen) – Wählen Sie diese Option, wenn Sie die Benutzer einer oder mehreren Gruppen zuordnen möchten, die bereits über Berechtigungsrichtlinien verfügen. IAM zeigt eine Liste der Gruppen in Ihrem Konto an, zusammen mit ihren angefügten Richtlinien. Sie können eine oder mehrere vorhandene Gruppen auswählen oder Create group (Gruppe erstellen) auswählen, um eine neue Gruppe zu erstellen. Weitere Informationen finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer.
   - Copy permissions (Berechtigungen kopieren) – Wählen Sie diese Option aus, um alle Gruppenmitgliedschaften, angefügten verwalteten Richtlinien und Inlinerichtlinien sowie alle vorhandenen Berechtigungsgrenzen von einem bestehenden Benutzer auf den neuen Benutzer zu kopieren. IAM zeigt eine Liste der Benutzer in Ihrem Konto an. Wählen Sie den Benutzer aus, dessen Berechtigungen am besten mit den Anforderungen Ihres neuen Benutzers übereinstimmen.
   - Richtlinien direkt anhängen — Wählen Sie diese Option, um eine Liste der AWS verwalteten und vom Kunden verwalteten Richtlinien in Ihrem Konto anzuzeigen. Wählen Sie die Richtlinien aus, die Sie dem Benutzer anfhängen möchten, oder wählen Sie Create policy (Richtlinie erstellen), um eine neue Registerkarte im Browser zu öffnen und eine neue Richtlinie zu erstellen. Weitere Informationen finden Sie im Schritt 4 der Anleitung Erstellen von IAM-Richtlinien. Nachdem Sie die Richtlinie erstellt haben, schließen Sie die Registerkarte und kehren zur ursprünglichen Registerkarte zurück, um die Richtlinie dem Benutzer hinzuzufügen.

   > **Tip**
   >
   > Ordnen Sie Ihre Richtlinien nach Möglichkeit einer Gruppe zu und machen die Benutzer zu Mitgliedern der entsprechenden Gruppen.

8. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature.

   Öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Sie Use a permissions boundary to control the maximum permissions (Eine Berechtigungsgrenze verwenden, um die maximalen Berechtigungen zu steuern) aus. IAM zeigt eine Liste der AWS verwalteten und vom Kunden verwalteten Richtlinien in Ihrem Konto an. Wählen Sie die Richtlinie

aus, die für die Berechtigungsgrenze verwendet werden soll, oder wählen Sie Create policy (Richtlinie erstellen) aus, um eine neue Registerkarte im Browser zu öffnen und eine neue Richtlinie zu erstellen. Weitere Informationen finden Sie im Schritt 4 der Anleitung Erstellen von IAM-Richtlinien. Nachdem Sie die Richtlinie erstellt haben, schließen Sie die Registerkarte und kehren zur ursprünglichen Registerkarte zurück, um die Richtlinie auszuwählen, die für die Berechtigungsgrenze verwendet werden soll.

9. Klicken Sie auf Weiter.
10. (Optional) Auf der Seite Review and create (Überprüfen und erstellen) wählen Sie unter Tags (Tags) die Option Add new tag (Neues Tag hinzufügen), um dem Benutzer Metadaten hinzuzufügen, indem Sie Tags als Schlüssel-Wert-Paare anhängen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
11. Überprüfen Sie alle Auswahlen, die Sie bis zu diesem Punkt getroffen haben. Wenn Sie bereit sind, fortzufahren, wählen Sie Create user (Benutzer erstellen) aus.
12. Rufen Sie auf der Seite Retrieve password (Passwort abrufen) das dem Benutzer zugewiesene Passwort ab:

    - Wählen Sie neben dem Passwort die Option Show (Anzeigen) aus, um das Passwort des Benutzers anzuzeigen, sodass Sie es manuell aufzeichnen können.
    - Wählen Sie Download .csv (CSV-Datei herunterladen) aus, um die Anmeldeinformationen des Benutzers als CSV-Datei herunterzuladen, die Sie an einem sicheren Ort speichern können.
13. Wählen Sie Email sign-in instructions (E-Mail-Anmeldeanweisungen) aus. Dadurch wird Ihr lokaler E-Mail-Client aufgerufen und sie können den E-Mail-Entwurf anpassen und an den Benutzer senden. Die E-Mail-Vorlage enthält die folgenden Details für jeden Benutzer:

    - Benutzername
    - URL der Anmelde-Website des Kontos. Verwenden Sie das folgende Beispiel und ersetzen Sie dabei die richtige Konto-ID-Nummer oder den Konto-Alias:

```
https://AWS-account-ID or alias.signin.aws.amazon.com/console
```

> **Important**
>
> Das Passwort des Benutzers ist nicht in der generierten E-Mail enthalten. Sie müssen dem Benutzer das Passwort in einer Form zukommen lassen, die den Sicherheitsrichtlinien Ihres Unternehmens entspricht.

14. Wenn der Benutzer auch Zugriffstasten für den programmatischen Zugriff benötigt, finden Sie weitere Informationen unter. Verwalten der Zugriffsschlüssel für IAM-Benutzer

Erstellen von IAM-Benutzern (AWS CLI)

Sie können den verwenden AWS CLI , um einen IAM-Benutzer zu erstellen.

So erstellen Sie einen IAM-Benutzer (AWS CLI)

1. Erstellen eines Benutzers.
   - aws iam create-user
2. (Optional) Geben Sie dem Benutzer Zugriff auf die AWS Management Console. Hierfür ist ein Passwort erforderlich. Sie müssen dem Benutzer auch die URL der Anmelde-Seite Ihres Kontos mitteilen.
   - war ich create-login-profile
3. (Optional) Geben Sie dem Benutzer programmgesteuerten Zugriff. Hierfür sind Zugriffsschlüssel erforderlich.
   - war ich create-access-key
   - Tools für Windows PowerShell: New-IAM AccessKey
   - IAM-API: CreateAccessKey

> **Important**
>
> Dies ist Ihre einzige Möglichkeit, die geheimen Zugriffsschlüssel einzusehen oder herunterzuladen. Sie müssen diese Informationen Ihren Benutzern zur Verfügung stellen, bevor sie die AWS API verwenden können. Speichern Sie die neue Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel des Benutzers an einem

sicheren Speicherort. Sie haben nach diesem Schritt keinen Zugriff mehr auf die geheimen Zugriffsschlüssel.

4. Fügen Sie den Benutzer zu einer oder mehreren Gruppen hinzu. Zu den von Ihnen angegebenen Gruppen sollten die Richtlinien angefügt sein, die dem Benutzer die entsprechenden Berechtigungen erteilen.

   - war ich add-user-to-group

5. (Optional) Fügen Sie dem Benutzer eine Richtlinie hinzu, die die Berechtigungen des Benutzers definiert. Hinweis: Zur Verwaltung der Benutzerberechtigungen empfehlen wir, den Benutzer zu einer Gruppe hinzuzufügen und zur Gruppe eine Richtlinie anzufügen, anstatt sie direkt dem Benutzer anzufügen.

   - war ich attach-user-policy

6. (Optional) Fügen Sie dem Benutzer benutzerdefinierte Attribute durch Zuweisen von Tags hinzu. Weitere Informationen finden Sie unter Verwaltung von Tags für IAM-Benutzer (AWS CLI oder AWS API).
7. (Optional) Erteilen Sie dem Benutzer die Berechtigung zum Verwalten ihrer eigenen Sicherheitsanmeldeinformationen. Weitere Informationen finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

IAM-Benutzer erstellen (API)AWS

Sie können die AWS API verwenden, um einen IAM-Benutzer zu erstellen.

Um einen IAM-Benutzer über die (API)AWS zu erstellen

1. Erstellen eines Benutzers.

   - CreateUser

2. (Optional) Geben Sie dem Benutzer Zugriff auf die AWS Management Console. Hierfür ist ein Passwort erforderlich. Sie müssen dem Benutzer auch die URL der Anmelde-Seite Ihres Kontos mitteilen.

   - CreateLoginProfile

3. (Optional) Geben Sie dem Benutzer programmgesteuerten Zugriff. Hierfür sind Zugriffsschlüssel erforderlich.

- CreateAccessKey

> **Important**
>
> Dies ist Ihre einzige Möglichkeit, die geheimen Zugriffsschlüssel einzusehen oder herunterzuladen. Sie müssen diese Informationen Ihren Benutzern zur Verfügung stellen, bevor sie die AWS API verwenden können. Speichern Sie die neue Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel des Benutzers an einem sicheren Speicherort. Sie haben nach diesem Schritt keinen Zugriff mehr auf die geheimen Zugriffsschlüssel.

4. Fügen Sie den Benutzer zu einer oder mehreren Gruppen hinzu. Zu den von Ihnen angegebenen Gruppen sollten die Richtlinien angefügt sein, die dem Benutzer die entsprechenden Berechtigungen erteilen.

   - AddUserToGroup

5. (Optional) Fügen Sie dem Benutzer eine Richtlinie hinzu, die die Berechtigungen des Benutzers definiert. Hinweis: Zur Verwaltung der Benutzerberechtigungen empfehlen wir, den Benutzer zu einer Gruppe hinzuzufügen und zur Gruppe eine Richtlinie anzufügen, anstatt sie direkt dem Benutzer anzufügen.

   - AttachUserPolicy

6. (Optional) Fügen Sie dem Benutzer benutzerdefinierte Attribute durch Zuweisen von Tags hinzu. Weitere Informationen finden Sie unter Verwaltung von Tags für IAM-Benutzer (AWS CLI oder AWS API).

7. (Optional) Erteilen Sie dem Benutzer die Berechtigung zum Verwalten ihrer eigenen Sicherheitsanmeldeinformationen. Weitere Informationen finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Steuern des IAM-Benutzerzugriffs auf die AWS Management Console

IAM-Benutzer mit entsprechender Genehmigung, die sich AWS-Konto über bei Ihnen anmelden, AWS Management Console können auf Ihre AWS Ressourcen zugreifen. Die folgende Liste zeigt, wie Sie IAM-Benutzern über die Zugriff auf Ihre AWS-Konto Ressourcen gewähren können. AWS

Management Console Außerdem wird gezeigt, wie IAM-Benutzer über die Website auf andere AWS Kontofunktionen zugreifen können. AWS

> **Note**
>
> Die Nutzung von IAM ist kostenlos.

Das AWS Management Console

Sie erstellen für jeden IAM-Benutzer, der Zugriff auf die AWS Management Console benötigt, ein Passwort. Benutzer greifen über Ihre IAM-fähige AWS-Konto Anmeldeseite auf die Konsole zu. Weitere Informationen zum Zugriff auf die Anmeldeseite finden Sie unter Anmelden bei AWS im AWS-Anmeldung -Benutzerhandbuch. Weitere Informationen zum Erstellen von Passwörtern finden Sie unter Benutzerpasswörter verwalten in AWS.

Sie können verhindern, dass ein IAM-Benutzer auf die zugreift, AWS Management Console indem Sie sein Passwort entfernen. Dadurch wird verhindert, dass sie sich AWS Management Console mit ihren Anmeldeinformationen bei der anmelden. Es ändert weder ihre Berechtigungen noch verhindert, dass sie mit einer angenommenen Rolle auf die Konsole zugreifen. Wenn der Benutzer über aktive Zugriffsschlüssel verfügt, funktionieren diese weiterhin und ermöglichen den Zugriff über die AWS CLI Tools für Windows PowerShell, die AWS API oder die AWS Console Mobile Application.

Ihre AWS Ressourcen, wie Amazon EC2 EC2-Instances, Amazon S3 S3-Buckets usw.

Selbst wenn Ihre IAM-Benutzer Passwörter haben, benötigen Sie dennoch die Berechtigung für den Zugriff auf Ihre AWS -Ressourcen. Ein neu erstellter IAM-Benutzer hat standardmäßig zunächst keine Berechtigungen. Um einem IAM-Benutzer die benötigten Berechtigungen zu gewähren, weisen Sie ihm Richtlinien zu. Wenn Sie viele IAM-Benutzer haben, die dieselben Aufgaben mit denselben Ressourcen ausführen, können Sie diese IAM-Benutzer einer Gruppe zuweisen. Weisen Sie dieser Gruppe dann die Berechtigungen zu. Weitere Informationen zum Erstellen von IAM-Benutzer und Gruppen finden Sie unter IAM-Identitäten (Benutzer, Gruppen und Rollen). Weitere Informationen zum Verwenden von Richtlinien, um Berechtigungen festzulegen, finden Sie unter Zugriffsmanagement für AWS Ressourcen.

AWS Diskussionsforen

Die Beiträge in den AWS -Diskussionsforen können von jedem gelesen werden. Benutzer, die Fragen oder Kommentare im AWS Diskussionsforum posten möchten, können dies mit ihrem

Benutzernamen tun. Wenn ein Benutzer zum ersten Mal Beiträge im AWS Diskussionsforum veröffentlicht, wird er aufgefordert, einen Spitznamen und eine E-Mail-Adresse einzugeben. Nur dieser Benutzer kann diesen Spitznamen in den AWS Diskussionsforen verwenden.

Ihre AWS-Konto Rechnungs- und Nutzungsinformationen

Sie können Benutzern Zugriff auf Ihre AWS-Konto Rechnungs- und Nutzungsinformationen gewähren. Weitere Informationen finden Sie im AWS Billing -Benutzerhandbuch unter Zugriff auf Ihre Rechnungsdaten kontrollieren.

Ihre AWS-Konto Profilinformationen

Benutzer können nicht auf Ihre AWS-Konto Profilinformationen zugreifen.

Ihre AWS-Konto Sicherheitsanmeldedaten

Benutzer können nicht auf Ihre AWS-Konto Sicherheitsanmeldedaten zugreifen.

> **Note**
>
> IAM-Richtlinien steuern den Zugriff unabhängig von der Schnittstelle. Beispielsweise können Sie einem Benutzer ein Passwort für den Zugriff auf bereitstellen AWS Management Console. Die Richtlinien für diesen Benutzer (oder alle Gruppen, zu denen der Benutzer gehört) würden steuern, welche Aktionen der Benutzer in der AWS Management Console ausführen kann. Oder Sie könnten dem Benutzer AWS Zugriffsschlüssel für API-Aufrufe zur Verfügung stellen AWS. Die Richtlinien würden steuern, welche Aktionen der Benutzer über eine Bibliothek oder einen Client aufrufen kann, die bzw. der diese Zugriffsschlüssel für die Authentifizierung verwendet.

So melden sich IAM-Benutzer an AWS

Um sich AWS Management Console als IAM-Benutzer anzumelden, müssen Sie zusätzlich zu Ihrem Benutzernamen und Passwort Ihre Konto-ID oder Ihren Kontoalias angeben. Wenn Ihr Administrator Ihren IAM-Benutzer in der Konsole erstellt, sollten sie Ihnen Ihre Anmeldeinformationen gesendet haben, einschließlich Ihres Benutzernamens und der URL zu Ihrer Kontoanmeldeseite, die Ihre Konto-ID oder Ihren Kontoalias enthält.

```
https://My_AWS_Account_ID.signin.aws.amazon.com/console/
```

> **Tipp**
>
> Um ein Lesezeichen für die Anmeldeseite Ihres Kontos in Ihrem Webbrowser zu erstellen, sollten Sie die Anmelde-URL für Ihr Konto manuell im Lesezeicheneintrag eingeben. Verwenden Sie keinen Webbrowser zum Anlegen von Lesezeichen, weil Weiterleitungen die Anmelde-URL verschleiern können.

Sie können sich auch auf dem folgenden Endpunkt für die allgemeine Anmeldung anmelden und Ihre Konto-ID oder den Kontopalias manuell eingeben:

```
https://console.aws.amazon.com/
```

Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich den IAM-Benutzernamen und die Kontoinformationen zu merken. Wenn der Benutzer das nächste Mal eine Seite in der aufruft AWS Management Console, verwendet die Konsole das Cookie, um den Benutzer auf die Anmeldeseite für das Konto weiterzuleiten.

Sie haben nur Zugriff auf die AWS Ressourcen, die Ihr Administrator in der Richtlinie festlegt, die mit Ihrer IAM-Benutzeridentität verknüpft ist. Um in der Konsole arbeiten zu können, müssen Sie über die Berechtigungen verfügen, um die von der Konsole ausgeführten Aktionen auszuführen, z. B. AWS Ressourcen aufzulisten und zu erstellen. Weitere Informationen finden Sie unter Zugriffsmanagement für AWS Ressourcen und Beispiele für identitätsbasierte Richtlinien in IAM.

> **Note**
>
> Wenn Ihr Unternehmen über ein Identitätssystem verfügt, können Sie auch eine Single Sign-On (SSO)-Option erstellen. SSO ermöglicht Benutzern den AWS Management Console Zugriff auf Ihr Konto, ohne dass sie über eine IAM-Benutzeridentität verfügen müssen. SSO macht es außerdem überflüssig, dass sich Benutzer auf der Website Ihrer Organisation anmelden und sich AWS separat anmelden müssen. Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

Anmeldedetails einloggen CloudTrail

Wenn Sie CloudTrail die Protokollierung von Anmeldeereignissen in Ihren Protokollen aktivieren, müssen Sie sich darüber im Klaren sein, wer CloudTrail auswählt, wo die Ereignisse protokolliert werden.

- Wenn sich Ihre Benutzer direkt bei einer Konsole anmelden, werden sie entweder zu einem globalen oder regionalen Anmeldeendpunkt umgeleitet, je nachdem, ob die ausgewählte Servicekonsole Regionen unterstützt. Die Startseite der Hauptkonsole beispielsweise unterstützt Regionen. Wenn Sie sich also bei der nachfolgenden URL anmelden,

  ```
  https://alias.signin.aws.amazon.com/console
  ```

  Sie werden zu einem regionalen Anmeldeendpunkt weitergeleitet`https://us-east-2.signin.aws.amazon.com`, z. B. was zu einem regionalen CloudTrail Protokolleintrag im Protokoll der Benutzerregion führt:

  Andererseits unterstützt die Amazon S3-Konsole keine Regionen. Wenn Sie sich also bei der folgenden URL anmelden,

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

  AWS leitet Sie zum globalen Anmeldeendpunkt unter weiter`https://signin.aws.amazon.com`, was zu einem globalen CloudTrail Protokolleintrag führt.
- Sie können manuell einen bestimmten regionalen Anmeldeendpunkt anfordern, indem Sie sich über eine URL-Syntax wie die folgende bei der Hauptkonsolen-Startseite mit Regionen anmelden:

  ```
  https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
  ```

  AWS leitet Sie zum `ap-southeast-1` regionalen Anmeldeendpunkt weiter und führt zu einem regionalen CloudTrail Protokollereignis.

Weitere Informationen zu CloudTrail und IAM finden Sie unter Protokollieren von IAM-Ereignissen mit. CloudTrail

Wenn Benutzer programmgesteuerten Zugriff benötigen, um mit Ihrem Konto zu arbeiten, können Sie für jeden ein Zugriffsschlüsselpaar (eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel) erstellen. Es gibt jedoch sicherere Alternativen, die Sie in Betracht ziehen sollten, bevor Sie

Zugriffsschlüssel für Benutzer erstellen. Weitere Informationen finden Sie unter Überlegungen und Alternativen für Schlüssel für den langfristigen Zugriff in der Allgemeine AWS-Referenz.

Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite

IAM-Benutzer, die für die Multi-Faktor-Authentifizierung (MFA)-Geräte konfiguriert worden sind, müssen ihre MFA-Geräte zum Anmelden bei der AWS Management Console verwenden. Nachdem der Benutzer seine Anmeldeinformationen eingegeben hat, AWS überprüft er das Konto des Benutzers, um festzustellen, ob MFA für diesen Benutzer erforderlich ist. Die folgenden Themen enthalten Informationen dazu, wie Benutzer die Anmeldung abschließen, wenn MFA erforderlich ist.

Themen

- Anmeldung mit mehreren aktivierten MFA-Geräten
- Anmelden mit einem FIDO-Sicherheitsschlüssel
- Anmelden mit einem virtuellen MFA-Gerät
- Anmelden mit einem Hardware-TOTP-Token

Anmeldung mit mehreren aktivierten MFA-Geräten

Wenn sich ein Benutzer AWS Management Console als AWS-Konto Root-Benutzer oder IAM-Benutzer anmeldet und mehrere MFA-Geräte für dieses Konto aktiviert sind, muss er sich nur mit einem MFA-Gerät anmelden. Nachdem sich der Benutzer mit dem Passwort des Benutzers authentifiziert hat, wählt er aus, welchen MFA-Gerätetyp er verwenden möchte, um die Authentifizierung abzuschließen. Anschließend wird der Benutzer aufgefordert, sich mit dem ausgewählten Gerätetyp zu authentifizieren.

Anmelden mit einem FIDO-Sicherheitsschlüssel

Wenn MFA für den Benutzer verlangt ist, wird eine zweite Anmeldeseite angezeigt. Der Benutzer muss auf den FIDO-Sicherheitsschlüssel tippen.

> **Note**
>
> Google-Chrome-Nutzer sollten keine der verfügbaren Optionen im Popup auswählen, in dem Sie zu Folgendem aufgefordert werden: Verifiy your identity with amazon.com (Verifizieren Sie Ihre Identität mit amazon.com). Sie müssen nur auf den Sicherheitsschlüssel tippen.

Im Gegensatz zu anderen MFA-Geräten können FIDO-Sicherheitsschlüssel ihre Synchronisation nicht verlieren. Administratoren können einen FIDO-Sicherheitsschlüssel deaktivieren, wenn er verloren geht oder beschädigt ist. Weitere Informationen finden Sie unter Deaktivieren von MFA-Geräten (Konsole).

Informationen zu Browsern, die dies unterstützen, WebAuthn und zu FIDO-kompatiblen Geräten, die dies unterstützen, finden Sie unter. AWS Unterstützte Konfigurationen für die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln

Anmelden mit einem virtuellen MFA-Gerät

Wenn MFA für den Benutzer verlangt ist, wird eine zweite Anmeldeseite angezeigt. Im Feld MFA code (MFA-Code) muss der Benutzer den von der MFA-Anwendung bereitgestellten Zahlencode eingeben.

Wenn der MFA-Code richtig ist, hat der Benutzer Zugriff auf die AWS Management Console. Wenn der Code falsch ist, kann der Benutzer es mit einem anderen Code erneut versuchen.

Ein virtuelles MFA-Gerät kann die Synchronisierung verlieren. Wenn sich ein Benutzer AWS Management Console nach mehreren Versuchen nicht bei der anmelden kann, wird er aufgefordert, das virtuelle MFA-Gerät zu synchronisieren. Die Benutzer können die Anweisungen auf dem Bildschirm befolgen, um das virtuelle MFA-Gerät zu synchronisieren. Informationen darüber, wie Sie ein Gerät im Namen eines Benutzers in Ihrem synchronisieren können AWS-Konto, finden Sie unterResynchronisieren von virtuellen und physischen MFA-Geräten.

Anmelden mit einem Hardware-TOTP-Token

Wenn MFA für den Benutzer verlangt ist, wird eine zweite Anmeldeseite angezeigt. Im Feld MFA code (MFA-Code) muss der Benutzer den vom Hardware-TOTP-Token bereitgestellten Zahlencode eingeben.

Wenn der MFA-Code richtig ist, hat der Benutzer Zugriff auf die AWS Management Console. Wenn der Code falsch ist, kann der Benutzer es mit einem anderen Code erneut versuchen.

Hardware-TOTP-Token können ihre Synchronisation verlieren. Wenn sich ein Benutzer AWS Management Console nach mehreren Versuchen nicht bei der anmelden kann, wird er aufgefordert, das MFA-Token-Gerät zu synchronisieren. Die Benutzer können die Anweisungen auf dem Bildschirm befolgen, um das MFA-Token-Gerät zu synchronisieren. Informationen darüber, wie Sie ein Gerät im Namen eines Benutzers in Ihrem synchronisieren können AWS-Konto, finden Sie unterResynchronisieren von virtuellen und physischen MFA-Geräten.

Verwalten von IAM-Benutzern

> **Note**
>
> Als bewährte Methode empfehlen wir, dass menschliche Benutzer für den Zugriff mit temporären Anmeldeinformationen einen Verbund mit einem Identitätsanbieter AWS verwenden müssen. Wenn Sie die bewährten Methoden befolgen, verwalten Sie keine IAM-Benutzer und -Gruppen. Stattdessen werden Ihre Benutzer und Gruppen außerhalb von AWS Ressourcen verwaltet AWS und können als föderierte Identität darauf zugreifen. Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter, dem AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe von Anmeldeinformationen, die über eine Identitätsquelle bereitgestellt wurden, auf AWS Dienste zugreift. Verbundidentitäten verwenden die von ihrem Identitätsanbieter definierten Gruppen. Wenn Sie dies verwenden AWS IAM Identity Center, finden Sie unter Identitäten in IAM Identity Center verwalten im AWS IAM Identity Center Benutzerhandbuch Informationen zum Erstellen von Benutzern und Gruppen in IAM Identity Center.

Amazon Web Services bietet mehrere Tools zum Verwalten von IAM-Benutzern in Ihrem AWS-Konto. Sie können die IAM-Benutzer in Ihrem Konto oder in einer Gruppe auflisten, oder alle Gruppen, in denen ein Benutzer Mitglied ist. Sie können den Pfad eines IAM-Benutzers umbenennen oder ändern. Wenn Sie dazu übergehen, Verbundidentitäten anstelle von IAM-Benutzern zu verwenden, können Sie einen IAM-Benutzer aus Ihrem AWS -Konto löschen oder den Benutzer deaktivieren.

Weitere Informationen zum Hinzufügen, Ändern oder Entfernen von verwalteten Richtlinien für einen IAM-Benutzer finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer. Weitere Informationen zum Verwalten von Inline-Richtlinien für IAM-Benutzer finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen, Bearbeiten von IAM-Richtlinien und Löschen von IAM-Richtlinien. Es hat sich bewährt, verwaltete Richtlinien anstelle von eingebundenen Richtlinien zu verwenden. Die AWS -verwalteten Richtlinien stellen Berechtigungen für viele häufige Anwendungsfälle bereit. Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie für alle Kunden verfügbar sind. AWS Daher empfehlen wir Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete Richtlinien definieren, die spezifisch auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS

verwaltete Richtlinien. Weitere Informationen zu AWS verwalteten Richtlinien, die für bestimmte Aufgabenbereiche konzipiert sind, finden Sie unter. AWS verwaltete Richtlinien für Jobfunktionen

Weitere Informationen zum Validieren von IAM-Richtlinien finden Sie unter Validieren von IAM-Richtlinien.

> **Tip**
>
> IAM Access Analyzer kann die Services und Aktionen analysieren, die Ihre IAM-Rollen verwenden, und generiert dann eine fein abgestimmte Richtlinie, die Sie verwenden können. Nachdem Sie jede generierte Richtlinie getestet haben, können Sie die Richtlinie in Ihrer Produktionsumgebung bereitstellen. Dadurch wird sichergestellt, dass Sie nur die erforderlichen Berechtigungen für Ihre Workloads gewähren. Weitere Informationen zur Richtlinienerstellung finden Sie unter IAM Access Analyzer Richtlinienerstellung.

Weitere Informationen zum Verwalten von IAM-Benutzerpasswörtern finden Sie unter Verwalten von Passwörtern für IAM-Benutzer,

Themen

Anzeigen des Benutzerzugriffs

Bevor Sie einen Benutzer löschen, sollten Sie seine kürzliche Service-Level-Aktivität überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Auflisten von IAM-Benutzern

Sie können die IAM-Benutzer in Ihrer AWS-Konto oder in einer bestimmten IAM-Benutzergruppe sowie alle Benutzergruppen auflisten, denen ein Benutzer angehört. Weitere Informationen zu den

Berechtigungen, die Sie zum Auflisten von Benutzern benötigen, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

So listen Sie alle Benutzer im Konto auf

- AWS Management Console: Wählen Sie im Navigationsbereich Benutzer. In der Konsole werden die Benutzer in Ihrer angezeigt. AWS-Konto
- AWS CLI: aws iam list-users
- AWS API: ListUsers

So listen Sie die Benutzer in einer bestimmten Gruppe auf

- AWS Management Console: Wählen Sie im Navigationsbereich Benutzergruppen, den Namen der Gruppe und anschließend die Registerkarte Benutzer.
- AWS CLI: aws iam get-group
- AWS API: GetGroup

So listen Sie alle Gruppen auf, denen ein Benutzer angehört

- AWS Management Console: Wählen Sie im Navigationsbereich Benutzer, den Benutzernamen und anschließend die Registerkarte Gruppen.
- AWS CLI: wie ich list-groups-for-user
- AWS API: ListGroupsForUser

Umbenennen eines IAM-Benutzers

Um den Namen oder Pfad eines Benutzers zu ändern, müssen Sie die AWS CLI Tools für Windows PowerShell oder die AWS API verwenden. In der Konsole gibt es keine Möglichkeit zum Umbenennen eines Benutzers. Weitere Informationen zu den Berechtigungen, die Sie zum Umbenennen eines Benutzers benötigen, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

Wenn Sie den Namen oder den Pfad eines Benutzers ändern, geschieht Folgendes:

- Alle Richtlinien, die dem Benutzer zugewiesen sind, verbleiben bei ihm unter dem neuen Namen.
- Der Benutzer verbleibt unter dem neuen Namen in den gleichen Gruppen.

- Die eindeutige ID des Benutzers bleibt unverändert. Weitere Informationen zu eindeutigen IDs finden Sie unter Eindeutige Bezeichner.
- Die Ressourcen- oder Rollenrichtlinien, die auf den Benutzer als Auftraggeber verweisen (dem Benutzer wird Zugriff gewährt), werden automatisch mit dem neuen Namen oder Pfad aktualisiert. Beispiel: Die warteschlangenbasierten Richtlinien in Amazon SQS oder die ressourcenbasierten Richtlinien in Amazon S3 werden automatisch mit dem neuen Namen und Pfad aktualisiert.

In IAM werden die Richtlinien, die auf den Benutzer als Ressource verweisen, nicht automatisch mit dem neuen Namen oder Pfad aktualisiert. Dies müssen Sie manuell vornehmen. Beispiel: Dem Benutzer `Richard` ist eine Richtlinie zugewiesen, mit der er seine Sicherheitsanmeldeinformationen verwalten kann. Wenn ein Administrator `Richard` in `Rich` umbenennt, muss er auch die Richtlinie aktualisieren, um die Ressource von dieser Zeichenfolge:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

in diese zu ändern:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

Dies gilt auch, wenn ein Administrator den Pfad ändert. Der Administrator muss die Richtlinie aktualisieren, damit der neue Pfad für den Benutzer wiedergegeben wird.

So benennen Sie einen Benutzer um

- AWS CLI: aws iam update-user
- AWS API: UpdateUser

Löschen eines IAM-Benutzers

Sie können einen IAM-Benutzer aus Ihrem löschen, AWS-Konto wenn dieser Benutzer Ihr Unternehmen verlässt. Wenn der Benutzer vorübergehend nicht da ist, können Sie den Zugriff des Benutzers deaktivieren, anstatt ihn wie in Deaktivieren eines IAM-Benutzers beschrieben aus dem Konto zu löschen.

Themen

- Löschen eines IAM-Benutzers (Konsole)
- Löschen eines IAM-Benutzers (AWS CLI)

Löschen eines IAM-Benutzers (Konsole)

Wenn Sie den verwenden AWS Management Console , um einen IAM-Benutzer zu löschen, löscht IAM automatisch die folgenden Informationen für Sie:

- Der Benutzer
- Sämtliche Gruppenmitgliedschaften, d. h. der Benutzer wird aus allen IAM-Gruppen entfernt, denen er angehörte
- Alle Passwörter des Benutzers
- Alle Zugriffsschlüssel des Benutzers
- Alle eingebundenen Richtlinien, die mit dem Benutzer integriert sind (Richtlinien, die einem Benutzer über Gruppenberechtigungen zugewiesen wurden, sind hiervon nicht betroffen)

> **Note**
>
> IAM entfernt alle verwalteten Richtlinien, die dem Benutzer zugeordnet sind, wenn Sie den Benutzer löschen, löscht jedoch keine verwalteten Richtlinien.

- Alle zugehörigen MFA-Geräte

So löschen Sie einen IAM-Benutzer (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. Wählen Sie im Navigationsbereich Users (Benutzer) aus und aktivieren Sie dann das Kontrollkästchen neben dem Benutzernamen, den Sie löschen möchten, nicht den Namen oder die Zeile selbst.
3. Wählen Sie oben auf der Seite Delete role (Rolle löschen).
4. Geben Sie im Bestätigungsdialogfeld den Benutzernamen in das Texteingabefeld ein, um das Löschen des Benutzers zu bestätigen. Wählen Sie Löschen.

Löschen eines IAM-Benutzers (AWS CLI)

Im AWS Management Console Gegensatz zu müssen Sie beim Löschen eines Benutzers mit dem AWS CLI die dem Benutzer zugewiesenen Elemente manuell löschen. Dieser Vorgang veranschaulicht den Prozess.

So löschen Sie einen Benutzer aus dem Konto (AWS CLI)

1. Löschen Sie das Passwort des Benutzers, sofern vorhanden.

   `aws iam delete-login-profile`

2. Löschen Sie die Zugriffsschlüssel des Benutzers, wenn der Benutzer über solche verfügt.

   `aws iam list-access-keys` (zum Auflisten der Zugriffsschlüssel des Benutzers) und `aws iam delete-access-key`

3. Löschen Sie das Signaturzertifikat des Benutzers. Beachten Sie, dass einmal gelöschte Sicherheitsanmeldeinformationen nicht wiederhergestellt werden können. Dieser Vorgang ist endgültig.

   `aws iam list-signing-certificates` (zum Auflisten der Signaturzertifikate des Benutzers) und `aws iam delete-signing-certificate`

4. Löschen Sie den öffentlichen SSH-Schlüssel des Benutzers, wenn der Benutzer über diesen verfügt.

   `aws iam list-ssh-public-keys` (zum Auflisten der öffentlichen SSH-Schlüssel des Benutzers) und `aws iam delete-ssh-public-key`

5. Löschen Sie die Git-Anmeldeinformationen des Benutzers.

   `aws iam list-service-specific-credentials` (zum Auflisten der Git-Anmeldeinformationen des Benutzers) und `aws iam delete-service-specific-credential`

6. Deaktivieren Sie die Multi-Factor Authentication (MFA), wenn für den Benutzer eine solche verwendet wird.

   `aws iam list-mfa-devices` (zum Auflisten der MFA-Geräte des Benutzers `aws iam deactivate-mfa-device` (zum Deaktivieren des Geräts) und `aws iam delete-virtual-mfa-device` (zum dauerhaften Löschen eines virtuellen MFA-Geräts)

7. Löschen Sie die eingebundenen Richtlinien des Benutzers.

   `aws iam list-user-policies` (zum Auflisten der eingebundenen Richtlinien des Benutzers) und `aws iam delete-user-policy` (zum Löschen der Richtlinie)

8. Heben Sie die Verknüpfung aller verwalteten Richtlinien mit dem Benutzer auf.

`aws iam list-attached-user-policies` (zum Auflisten der verwalteten Richtlinien, die dem Benutzer zugeordnet sind) und `aws iam detach-user-policy` (zum Aufheben der Verknüpfung der Richtlinien)

9. Entfernen Sie den Benutzer aus allen Gruppen.

   `aws iam list-groups-for-user` (zum Auflisten der Gruppen, denen der Benutzer angehört) und `aws iam remove-user-from-group`

10. Löschen Sie den Benutzer.

    `aws iam delete-user`

Deaktivieren eines IAM-Benutzers

Möglicherweise müssen Sie IAM-Benutzer deaktivieren, während sie sich vorübergehend nicht in Ihrem Unternehmen befinden. Sie können ihre IAM-Benutzeranmeldedaten unverändert lassen und ihren AWS Zugriff trotzdem blockieren.

Um einen Benutzer zu deaktivieren, erstellen Sie eine Richtlinie und fügen Sie sie hinzu, um dem Benutzer den Zugriff auf AWS zu verweigern. Sie können den Zugriff des Benutzers später wiederherstellen.

Im Folgenden finden Sie zwei Beispiele für Verweigerungsrichtlinien, die Sie einem Benutzer zuordnen können, um ihm den Zugriff zu verweigern.

Die folgende Richtlinie beinhaltet keine zeitliche Begrenzung. Sie müssen die Richtlinie entfernen, um den Zugriff des Benutzers wiederherzustellen.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

Die folgende Richtlinie beinhaltet eine Bedingung, die die Richtlinie am 24. Dezember 2024 um 23:59 Uhr (UTC) startet und am 28. Februar 2025 um 23:59 Uhr (UTC) beendet.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

Ändern von Berechtigungen für einen IAM-Benutzer

Sie können die Berechtigungen für einen IAM-Benutzer in Ihrem ändern, AWS-Konto indem Sie dessen Gruppenmitgliedschaften ändern, die Berechtigungen eines vorhandenen Benutzers kopieren, Richtlinien direkt an einen Benutzer anhängen oder indem Sie eine Berechtigungsgrenze festlegen. Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die ein Benutzer haben kann. Bei Berechtigungsgrenzen handelt es sich um eine erweiterte Funktion. AWS

Weitere Informationen über die erforderlichen Berechtigungen, um die Berechtigungen eines Benutzers zu bearbeiten, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

Themen

- Anzeigen des Benutzerzugriffs
- Generieren einer Richtlinie basierend auf der Zugriffsaktivität eines Benutzers
- Hinzufügen von Berechtigungen für einem Benutzer (Konsole)
- Ändern von Berechtigungen für einen Benutzer (Konsole)
- Entfernen einer Berechtigungsrichtlinie von einem Benutzer (Konsole)
- Entfernen der Berechtigungsgrenze von einem Benutzer (Konsole)
- Berechtigungen (AWS CLI oder AWS API) eines Benutzers hinzufügen und entfernen

Anzeigen des Benutzerzugriffs

Bevor Sie die Berechtigungen für einen Benutzer ändern, sollten Sie seine kürzliche Service-Level-Aktivität überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Generieren einer Richtlinie basierend auf der Zugriffsaktivität eines Benutzers

Manchmal können Sie einer IAM-Entität (Benutzer oder Rolle) Berechtigungen erteilen, die über das hinausgehen, was diese benötigen. Um Ihnen beim Verfeinern der Berechtigungen zu helfen, können Sie eine IAM-Richtlinie generieren, die auf der Zugriffsaktivität für eine Entity basiert. IAM Access Analyzer überprüft Ihre AWS CloudTrail Protokolle und generiert eine Richtlinienvorlage, die die Berechtigungen enthält, die von der Entität in dem von Ihnen angegebenen Zeitraum verwendet wurden. Sie können die Vorlage verwenden, um eine verwaltete Richtlinie mit definierten Berechtigungen zu erstellen und sie dann an die IAM-Rolle anzuhängen. Auf diese Weise gewähren Sie nur die Berechtigungen, die der Benutzer oder die Rolle benötigt, um mit AWS Ressourcen für Ihren speziellen Anwendungsfall zu interagieren. Weitere Informationen hierzu finden Sie unter Generieren von Richtlinien basierend auf Zugriffsaktivitäten.

Hinzufügen von Berechtigungen für einem Benutzer (Konsole)

IAM bietet drei Möglichkeiten zum Hinzufügen von Berechtigungsrichtlinien zu einem Benutzer:

- Hinzufügen von Benutzern zu Gruppen – Machen Sie den Benutzer zu einem Mitglied einer Gruppe. Die Richtlinien der Gruppe werden dem Benutzer zugeordnet.
- Kopieren von Berechtigungen von vorhandenen Benutzern – Kopieren Sie alle Gruppenmitgliedschaften, angefügte verwaltete Richtlinien, eingebundene Richtlinien sowie vorhandene Berechtigungsgrenzen des Quellbenutzers.
- Anfügen von Richtlinien direkt zu Benutzern – Fügen Sie eine verwaltete Richtlinie direkt an den Benutzer an. Um die Verwaltung von Berechtigungen zu vereinfachen, ordnen Sie Ihre Richtlinien einer Gruppe zu und machen die Benutzer zu Mitgliedern der entsprechenden Gruppen.

> **Important**
>
> Wenn der Benutzer eine Berechtigungsgrenze hat, können Sie einem Benutzer nicht mehr Berechtigungen hinzufügen, als laut der Berechtigungsgrenze erlaubt sind.

Hinzufügen von Berechtigungen durch Hinzufügen des Benutzers zu einer Gruppe

Das Hinzufügen eines Benutzers zu einer Gruppe wirkt sich sofort auf den Benutzer.

So erteilen Sie einem Benutzer Berechtigungen, indem sie ihn zu einer Gruppe hinzufügen

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 auf Users (Benutzer).
3. Überprüfen Sie in der Konsole in der Spalte Groups (Gruppen) die aktuellen Gruppenmitgliedschaften der Benutzer. Falls erforderlich, fügen Sie die Spalte zur Tabelle der Benutzer hinzu, indem Sie die folgenden Schritte ausführen:

   1. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ( ).
   2. Klicken Sie im Dialogfeld auf Manage Columns (Spalten verwalten) und wählen Sie die Spalte Groups (Gruppen). Optional können Sie auch die Markierung der Kontrollkästchen für alle Spaltenüberschriften entfernen, die nicht in der Tabelle der Benutzer erscheinen sollen.
   3. Klicken Sie auf Close (Schließen), um zur Liste der Benutzer zurückzukehren.

   In der Spalte Groups (Gruppen) wird angegeben, zu welchen Gruppen der Benutzer gehört. Die Spalte enthält die Gruppennamen für bis zu zwei Gruppen. Wenn der Benutzer ein Mitglied von drei oder mehreren Gruppen ist, werden die ersten beiden Gruppen (alphabetisch geordnet) sowie die Anzahl der weiteren Gruppenmitgliedschaften angezeigt. Beispiel: Wenn der Benutzer zur Gruppe A, Gruppe B, Gruppe C und Gruppe D gehört, enthält das Feld den Wert Group A, Group B + 2 more. Um die Gesamtanzahl der Gruppenmitgliedschaften des Benutzers anzuzeigen, können Sie die Spalte Group count (Gruppenanzahl) zur Benutzertabelle hinzufügen.
4. Wählen Sie den Namen des Benutzers aus, dessen Berechtigungen Sie ändern möchten.

5. Wählen Sie die Registerkarte Permissions (Berechtigungen) und anschließend die Option Add Permissions (Berechtigung hinzufügen). Wählen Sie Add user to group.
6. Aktivieren Sie das Kontrollkästchen für jede Gruppe, die der Benutzer angehören soll. Die Liste enthält die Gruppennamen und Richtlinien, die dem Benutzer zugewiesen werden, wenn er Mitglied der Gruppe wird.
7. (Optional) Zusätzlich zu den vorhandenen Gruppen können Sie Create group (Gruppe erstellen) auswählen, um eine neue Gruppe zu definieren:

   a. Geben Sie auf der neuen Registerkarte für User group Name (Benutzergruppenname) den Namen für Ihre neue Gruppe ein.

   > **Note**
   >
   > Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS. Gruppennamen können eine Kombination aus bis zu 128 Buchstaben, Ziffern und die folgenden Zeichen enthalten: (+), Gleichheitszeichen (=), Komma (,), Punkt (.), at-Zeichen (@) und Bindestrich (-). Namen müssen innerhalb eines Kontos eindeutig sein. Es wird hierbei nicht zwischen Groß- und Kleinschreibung unterschieden. Sie können beispielsweise nicht zwei Gruppen mit dem Namen TESTGRUPPE und testgruppe erstellen.

   b. Markieren Sie ein oder mehrere Kontrollkästchen für die verwalteten Richtlinien, die Sie der Gruppe zuordnen möchten. Sie können auch anhand von Create policy (Richtlinie erstellen) eine neue verwaltete Richtlinie erstellen. Wenn Sie auf diese Weise vorgehen möchten, kehren Sie zu dieser Browser-Registerkarte oder zu diesem Fenster zurück. Wenn die neue Richtlinie erstellt wurde, wählen Sie erst Refresh (Aktualisieren) und dann die neue Richtlinie aus, die Sie Ihrer Gruppe anfügen möchten. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien.

   c. Wählen Sie Create user group (Benutzergruppe erstellen).

   d. Kehren Sie zur ursprünglichen Registerkarte zurück und aktualisieren Sie die Liste der Gruppen. Dann aktivieren Sie das Kontrollkästchen für die neue Gruppe.
8. Wählen Sie Next (Weiter) aus, um eine Liste der Gruppenmitgliedschaften anzuzeigen, die zu dem neuen Benutzer hinzugefügt werden soll. Wählen Sie dann Add permissions (Berechtigungen hinzufügen).

Hinzufügen von Berechtigungen durch Kopieren von einem anderen Benutzer

Das Kopieren der Berechtigungen wirkt sich sofort auf den Benutzer.

So fügen Sie Berechtigungen für einen Benutzer durch das Kopieren von Berechtigungen von einem anderen Benutzer hinzu

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 erst den Navigationsbereich Users (Benutzer) und dann den Namen des Benutzers aus, dessen Berechtigungen Sie ändern möchten, und klicken Sie dann auf die Registerkarte Permissions (Berechtigungen).
3. Wählen Sie Add permissions (Berechtigungen hinzufügen) und dann Copy permissions from existing user (Berechtigungen von vorhandenen Benutzern kopieren). Die Liste zeigt die verfügbaren Benutzer zusammen mit ihren Gruppenmitgliedschaften und zugeordneten Richtlinien an. Wenn die vollständige Liste der Gruppen oder Richtlinien nicht in eine Zeile passt, können Sie den Link and ***n*** more anklicken. Auf diese Weise öffnet sich eine neue Registerkarte und es wird die vollständige Liste der Richtlinien (Registerkarte Permissions) und Gruppen (Registerkarte Groups) angezeigt.
4. Aktivieren Sie das Ankreuzfeld neben dem Benutzer, dessen Berechtigungen Sie kopieren möchten.
5. Wählen Sie Next (Weiter) aus, um eine Liste der Änderungen für den Benutzer anzuzeigen. Wählen Sie dann Add permissions (Berechtigungen hinzufügen).

Hinzufügen von Berechtigungen durch direktes Anfügen von Richtlinien zum Benutzer

Das Hinzufügen von Richtlinien wirkt sich sofort auf den Benutzer.

So fügen Sie Berechtigungen durch direktes Anfügen von Richtlinien dem Benutzer hinzu

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 erst den Navigationsbereich Users (Benutzer) und dann den Namen des Benutzers aus, dessen Berechtigungen Sie ändern möchten, und klicken Sie dann auf die Registerkarte Permissions (Berechtigungen).
3. Wählen Sie Add permissions (Berechtigungen hinzufügen) aus und wählen Sie dann Attach policies directly (Richtlinien direkt anhängen) aus.

4. Markieren Sie ein oder mehrere Kontrollkästchen für die verwalteten Richtlinien, die Sie dem Benutzer zuordnen möchten. Sie können auch anhand von Create policy (Richtlinie erstellen) eine neue verwaltete Richtlinie erstellen. Wenn Sie das tun, kehren Sie zu dieser Browser-Registerkarte oder zu diesem Fenster zurück, wenn die neue Richtlinie erstellt worden ist. Klicken Sie erst auf Refresh (Aktualisieren) und dann auf das Kontrollkästchen der neuen Richtlinie, um sie ihrem Benutzer anzufügen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien.
5. Wählen Sie Next (Weiter) aus, um eine Liste der Richtlinien anzuzeigen, die dem Benutzer anzufügen sind. Wählen Sie dann Add permissions (Berechtigungen hinzufügen).

Die Berechtigungsgrenze für einen Benutzer festlegen

Das Festlegen einer Berechtigungsgrenze wirkt sich sofort auf den Benutzer.

Die Berechtigungsgrenze für einen Benutzer festlegen

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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Berechtigungsgrenze Sie ändern möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen). Falls erforderlich, öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Sie Set permissions boundary (Berechtigungsgrenze festlegen).
5. Wählen Sie die Richtlinie aus, die Sie für die Berechtigungsgrenze verwenden möchten.
6. Wählen Sie Set boundary (Grenze festlegen).

Ändern von Berechtigungen für einen Benutzer (Konsole)

IAM ermöglicht Ihnen, die Berechtigungen, die einem Benutzer zugeordnet sind, auf folgende Weise zu ändern:

- Bearbeiten einer Berechtigungsrichtlinie – Bearbeiten einer Inline-Richtlinie des Benutzers, der eingebundenen Richtlinie der Gruppe des Benutzers, oder bearbeiten einer verwalteten Richtlinie, die dem Benutzer direkt oder aus einer Gruppe angefügt wird. Wenn der Benutzer eine Berechtigungsgrenze hat, können Sie nicht mehr Berechtigungen bereitstellen, als von der Richtlinie, die als Berechtigungsgrenze des Benutzers verwendet wurde, erlaubt ist.

- Ändern der Berechtigungsgrenze – Ändern Sie die Richtlinie, die als Berechtigungsgrenze für den Benutzer verwendet wird. Dadurch können die maximalen Berechtigungen, die ein Benutzer haben kann, erweitert oder eingeschränkt werden.

Bearbeiten einer Berechtigungsrichtlinie, die einem Benutzer hinzugefügt wurde

Eine Änderung der Berechtigungen wirkt sich sofort auf den Benutzer aus.

Die einem Benutzer hinzugefügten verwalteten Richtlinien bearbeiten

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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Berechtigungsrichtlinie Sie ändern möchten.
4. Wählen Sie die Registerkarte Berechtigungen. Falls erforderlich, öffnen Sie den Abschnitt Permissions policies (Berechtigungsrichtlinien) .
5. Wählen Sie den Namen der Richtlinie, die Sie bearbeiten möchten, um Details zur Richtlinie anzuzeigen. Wählen Sie die Registerkarte Policy usage (Richtliniennutzung), um andere Entitäten anzuzeigen, die möglicherweise beeinträchtigt werden, wenn Sie die Richtlinie bearbeiten.
6. Wählen Sie die Registerkarte Permissions (Berechtigungen) und überprüfen Sie die Berechtigungen, die durch die Richtlinie erteilt werden. Wählen Sie dann Edit policy (Richtlinie bearbeiten).
7. Bearbeiten Sie die Richtlinie und lösen Sie alle Empfehlungen zur policy validation (Richtlinienvalidierung). Weitere Informationen finden Sie unter Bearbeiten von IAM-Richtlinien.
8. Wählen Sie Review policy (Richtlinie überprüfen), überprüfen Sie die Richtlinienübersicht, und wählen Sie dann Save changes (Änderungen speichern).

Ändern der Berechtigungsgrenze für einen Benutzer

Das Ändern einer Berechtigungsgrenze wirkt sich sofort auf den Benutzer.

Ändern der Richtlinie zum Festlegen der Berechtigungsgrenze für einen Benutzer

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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Berechtigungsgrenze Sie ändern möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen). Falls erforderlich, öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Change boundary (Grenze ändern).
5. Wählen Sie die Richtlinie aus, die Sie für die Berechtigungsgrenze verwenden möchten.
6. Wählen Sie Set boundary (Grenze festlegen).

Entfernen einer Berechtigungsrichtlinie von einem Benutzer (Konsole)

Das Entfernen einer Richtlinie wirkt sich sofort auf den Benutzer.

So entfernen Sie Berechtigungen für IAM-Benutzer

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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Berechtigungsgrenze Sie entfernen möchten.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wenn Sie Berechtigungen durch Entfernen einer vorhandenen Richtlinie entfernen möchten, sehen Sie sich den Type (Typ) an, um zu verstehen, wie dem Benutzer die Richtlinie zugewiesen wird, bevor Sie Remove (Entfernen) zum Entfernen der Richtlinie auswählen:

   - Erfolgt die Anwendung der Richtlinie wegen einer Gruppenmitgliedschaft, wird mit Remove (Entfernen) der Benutzer aus der Gruppe entfernt. Denken Sie daran, dass einer Gruppe möglicherweise mehrere Richtlinien angefügt wurden. Wenn Sie einen Benutzer aus einer Gruppe entfernen, verliert er den Zugriff auf alle Richtlinien, die ihm über die Gruppenmitgliedschaft zugewiesen wurden.
   - Wenn die Richtlinie eine direkt zum Benutzer angefügte verwaltete Richtlinie ist, wird durch die Auswahl von Remove (Entfernen) die Richtlinie vom Benutzer getrennt. Dies hat keine Auswirkungen auf die Richtlinie selbst oder eine andere Entität, zu der die Richtlinie angefügt ist.

- Wenn die Richtlinie eine eingebettete Inlinerichtlinie ist, dann wird durch die Auswahl von X die Richtlinie aus IAM entfernt. Inlinerichtlinien, die direkt einem Benutzer angefügt werden, sind nur für diesen Benutzer vorhanden.

Entfernen der Berechtigungsgrenze von einem Benutzer (Konsole)

Das Entfernen einer Berechtigungsgrenze wirkt sich sofort auf den Benutzer.

Entfernen der Berechtigungsgrenze von einem Benutzer

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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Berechtigungsgrenze Sie entfernen möchten.
4. Wählen Sie die Registerkarte Berechtigungen. Falls erforderlich, öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Remove boundary (Grenze entfernen).
5. Wählen Sie Remove boundary (Grenze entfernen), um zu bestätigen, dass Sie die Berechtigungsgrenze entfernen möchten.

Berechtigungen (AWS CLI oder AWS API) eines Benutzers hinzufügen und entfernen

Um Berechtigungen programmgesteuert hinzuzufügen oder zu entfernen, müssen Sie die Gruppenmitgliedschaften hinzufügen oder entfernen, die verwaltete Richtlinien zuordnen bzw. diese Zuordnung aufheben oder die Inlinerichtlinien hinzufügen oder löschen. Weitere Informationen finden Sie unter den folgenden Themen:

- Hinzufügen und Entfernen von Benutzern in einer IAM-Gruppe
- Hinzufügen und Entfernen von IAM-Identitätsberechtigungen

Benutzerpasswörter verwalten in AWS

Sie können Passwörter für IAM-Benutzer in Ihrem Konto verwalten. IAM-Benutzer benötigen Passwörter, um auf die AWS Management Console zugreifen zu können. Benutzer benötigen

keine Passwörter, um mithilfe der Tools für Windows AWS CLI PowerShell, der AWS SDKs oder APIs programmgesteuert auf AWS Ressourcen zuzugreifen. Für diese Umgebungen haben Sie die Möglichkeit, IAM-Benutzern Zugriffsschlüssel zuzuweisen. Es gibt jedoch andere sicherere Alternativen als Zugangsschlüssel, die Sie zunächst in Betracht ziehen sollten. Weitere Informationen finden Sie unter AWS Sicherheitsnachweise.

Inhalt

Einrichten einer Kontopasswortrichtlinie für IAM-Benutzer

Sie können eine benutzerdefinierte Passwortrichtlinie für Sie einrichten AWS-Konto , um Komplexitätsanforderungen und obligatorische Rotationsperioden für die Passwörter Ihrer IAM-Benutzer festzulegen. Wenn Sie keine benutzerdefinierte Kennwortrichtlinie festlegen, müssen IAM-Benutzerkennwörter der AWS Standard-Passwortrichtlinie entsprechen. Weitere Informationen finden Sie unter Benutzerdefinierte Optionen für Passwortrichtlinien.

Themen

Einrichten einer Passwortrichtlinie

Die IAM-Passwortrichtlinie gilt nicht für das Root-Benutzer des AWS-Kontos Passwort oder die IAM-Benutzerzugriffsschlüssel. Wenn ein Passwort abläuft, kann sich der IAM-Benutzer nicht bei dem anmelden, AWS Management Console sondern seine Zugangsschlüssel weiterhin verwenden.

Wenn Sie eine Passwortrichtlinie erstellen oder ändern, werden die meisten Einstellungen darin wirksam, sobald die Benutzer ihre Passwörter ändern. Einige Einstellungen werden jedoch sofort wirksam. Zum Beispiel:

- Wenn sich die Anforderungen für die Mindestlänge oder Zeichenanforderungen ändern, werden diese Einstellungen bei der nächsten Änderung eines Passworts umgesetzt. Benutzer müssen ihre vorhandenen Passwörter nicht ändern, auch wenn diese nicht den Anforderungen der aktualisierten Passwortrichtlinie entsprechen.
- Wenn Sie einen Ablaufzeitraum für Passwörter festlegen, wird dieser sofort umgesetzt. Beispiel: Sie legen einen Passwort-Ablaufzeitraum von 90 Tagen fest. In diesem Fall läuft das Passwort für alle IAM-Benutzer ab, deren bestehendes Passwort älter als 90 Tage ist. Diese Benutzer müssen ihr Passwort bei der nächsten Anmeldung ändern.

Sie können keine „Lockout-Richtlinie“ erstellen, um einen Benutzer nach einer bestimmten Anzahl von fehlgeschlagenen Anmeldeversuchen aus dem Konto zu sperren. Zur Erhöhung der Sicherheit empfehlen wir Ihnen, eine starke Passwortrichtlinie mit Multi-Factor-Authentication (MFA) zu kombinieren. Weitere Informationen zu MFA finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

Zum Festlegen einer Passwortrichtlinie erforderliche Berechtigungen

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rolle) ihre KontoPasswortrichtlinie anzeigen oder bearbeiten kann. Sie können die folgenden Passwortrichtlinienaktionen in eine IAM-Richtlinie aufnehmen:

- `iam:GetAccountPasswordPolicy` – Ermöglicht der Entität das Anzeigen der Passwortrichtlinie für ihr Konto
- `iam:DeleteAccountPasswordPolicy` – Ermöglicht der Entität, die benutzerdefinierte Passwortrichtlinie für ihr Konto zu löschen und zur StandardPasswortrichtlinie zurückzukehren
- `iam:UpdateAccountPasswordPolicy` – Ermöglicht der Entität das Erstellen oder Ändern der benutzerdefinierten Passwortrichtlinie für ihr Konto

Die folgende Richtlinie ermöglicht den vollständigen Zugriff zum Anzeigen und Bearbeiten der KontoPasswortrichtlinie. Informationen zum Erstellen einer IAM-Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Informationen zu den Berechtigungen, die ein IAM-Benutzer zum Ändern seines eigenen Passworts benötigt, finden Sie unter Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können.

Standard-Passwortrichtlinie

Wenn ein Administrator keine benutzerdefinierte Kennwortrichtlinie festlegt, müssen die IAM-Benutzerkennwörter der AWS Standard-Passwortrichtlinie entsprechen.

Die StandardPasswortrichtlinie setzt die folgenden Bedingungen durch:

- Mindestpasswortlänge von 8 Zeichen und eine Maximallänge von 128 Zeichen
- Mindestens drei der folgenden Zeichenarten: Großbuchstaben, Kleinbuchstaben, Zahlen und die Symbole `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Sie müssen nicht mit Ihrem AWS-Konto Namen oder Ihrer E-Mail-Adresse identisch sein
- Passwort niemals ablaufen lassen

Benutzerdefinierte Optionen für Passwortrichtlinien

Wenn Sie eine benutzerdefinierte Passwortrichtlinie für Ihr Konto konfigurieren, können Sie die folgenden Bedingungen festlegen:

- Passwort-Mindestlänge – Sie können mindestens 6 Zeichen und maximal 128 Zeichen angeben.
- Passwortstärke – Sie können eines der folgenden Kontrollkästchen aktivieren, um die Stärke Ihrer IAM-BenutzerPasswörter zu definieren:

  - Erfordern mindestens einen Großbuchstaben aus dem lateinischen Alphabet (A–Z)
  - Erfordern mindestens einen Kleinbuchstaben aus dem lateinischen Alphabet (a–z)
  - Mindestens eine Zahl
  - Erfordert mindestens ein nicht alphanumerisches Zeichen `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Passwortablauf aktivieren – Sie können für die Gültigkeitsdauer von IAM-Benutzerpasswörtern mindestens 1 und maximal 1 095 Tage auswählen, die Passwörter nach ihrer Erstellung gültig sein sollen. Wenn Sie beispielsweise einen Ablaufzeitraum von 90 Tagen angeben, wirkt sich dies sofort auf alle Ihre Benutzer aus. Für Benutzer mit Passwörtern, die älter als 90 Tage sind, müssen sie, wenn sie sich nach der Änderung bei der Konsole anmelden, ein neues Passwort festlegen. Benutzer mit Passwörtern, die zwischen 75 und 89 Tage alt sind, erhalten eine AWS Management Console Warnung, dass ihr Passwort abläuft. IAM-Benutzer können ihr Passwort jederzeit ändern, wenn sie über eine entsprechende Berechtigung verfügen. Der Ablaufzeitraum beginnt von vorn, sobald ein neues Passwort festgelegt wird. Ein IAM-Benutzer kann immer nur ein gültiges Passwort gleichzeitig haben.
- Für den Ablauf des Kennworts ist ein Zurücksetzen durch den Administrator erforderlich — Wählen Sie diese Option, um AWS Management Console zu verhindern, dass IAM-Benutzer nach Ablauf des Kennworts ihre eigenen Passwörter aktualisieren. Stellen Sie vor dem Aktivieren dieser Option sicher, dass für das AWS-Konto mehr als ein Benutzer mit Administratorberechtigungen (also mit der Berechtigung zum Zurücksetzen von IAM-Benutzerpasswörtern) existiert. Administratoren mit `iam:UpdateLoginProfile`-Berechtigung können IAM-Benutzer-Passwörter zurücksetzen. IAM-Benutzer mit `iam:ChangePassword`-Berechtigung und aktive Zugriffsschlüssel können ihr eigenes IAM-Benutzer-Konsolenpasswort programmgesteuert zurücksetzen. Wenn Sie dieses Kontrollkästchen deaktivieren, müssen IAM-Benutzer mit abgelaufenen Passwörtern dennoch ein neues Passwort festlegen, bevor sie auf die AWS Management Console.
- Benutzern erlauben, ihr eigenes Passwort zu ändern – Sie können allen IAM-Benutzern in Ihrem Konto die Berechtigung gewähren ihr eigenes Passwort zu ändern. Dadurch erhalten Benutzer Zugriff auf die `iam:ChangePassword`-Aktion nur für ihren Benutzer und auf die `iam:GetAccountPasswordPolicy`-Aktion. Mit dieser Option wird nicht jedem Benutzer eine Berechtigungsrichtlinie zugewiesen. Stattdessen wendet IAM die Berechtigungen auf Kontoebene für alle Benutzer an. Sie können alternativ auch nur ausgewählten Benutzern die Berechtigung zum Verwalten ihrer eigenen Passwörter gewähren. Deaktivieren Sie dazu dieses Kontrollkästchen. Weitere Informationen dazu, wie Sie mithilfe von Richtlinien steuern, welche Benutzer Passwörter verwalten können, finden Sie unter Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können.

- Wiederverwendung von Passwörtern verhindern – Sie können verhindern, dass IAM-Benutzer eine bestimmte Anzahl an zuvor verwendeten Passwörtern wiederverwenden. Sie können eine Mindestanzahl von 1 und eine maximale Anzahl von 24 vorherigen Passwörtern angeben, die nicht wiederholt werden können.

Einrichten einer Passwortrichtlinie (Konsole)

Sie können die verwenden, AWS Management Console um eine benutzerdefinierte Kennwortrichtlinie zu erstellen, zu ändern oder zu löschen.

So erstellen Sie eine benutzerdefinierte Passwortrichtlinie (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. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).
3. Wählen Sie im Abschnitt Password policy (Passwortrichtlinie) die Option Edit (Bearbeiten) aus.
4. Wählen Sie Custom (Benutzerdefiniert), um eine benutzerdefinierte Passwortrichtlinie zu verwenden.
5. Wählen Sie die Optionen aus, die Sie auf Ihre Passwortrichtlinie anwenden möchten, und wählen Sie Änderungen speichern.
6. Bestätigen Sie, dass Sie die Richtlinie für benutzerdefinierte Passwörter festlegen möchten, indem Sie Set custom (Benutzerdefiniert festlegen) wählen.

So ändern Sie eine benutzerdefinierte Passwortrichtlinie (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. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).
3. Wählen Sie im Abschnitt Password policy (Passwortrichtlinie) die Option Edit (Bearbeiten) aus.
4. Wählen Sie die Optionen aus, die Sie auf Ihre Passwortrichtlinie anwenden möchten, und wählen Sie Änderungen speichern.
5. Bestätigen Sie, dass Sie die Richtlinie für benutzerdefinierte Passwörter festlegen möchten, indem Sie Set custom (Benutzerdefiniert festlegen) wählen.

So löschen Sie eine benutzerdefinierte Passwortrichtlinie (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. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).
3. Wählen Sie im Abschnitt Password policy (Passwortrichtlinie) die Option Edit (Bearbeiten) aus.
4. Wählen Sie IAM default (IAM-Standard), um die benutzerdefinierte Passwortrichtlinie zu löschen, und wählen Sie Save changes (Änderungen speichern).
5. Bestätigen Sie, dass Sie die IAM-Standardrichtlinie für Passwörter festlegen möchten, indem Sie Set default (Benutzerdefiniert festlegen) wählen.

Einrichten einer Passwortrichtlinie (AWS CLI)

Sie können den verwenden AWS Command Line Interface , um eine Passwortrichtlinie festzulegen.

Um die Passwortrichtlinie für benutzerdefinierte Konten über das zu verwalten AWS CLI

Führen Sie die folgenden Befehle aus:

- So erstellen oder ändern Sie die Richtlinie für benutzerdefinierte Passwörter: `aws iam update-account-password-policy`
- So zeigen Sie die Passwortrichtlinie an: `aws iam get-account-password-policy`
- So löschen Sie die Richtlinie für benutzerdefinierte Passwörter: `aws iam delete-account-password-policy`

Einrichtung einer Passwortrichtlinie (AWS API)

Sie können AWS API-Operationen verwenden, um eine Passwortrichtlinie festzulegen.

Um die Passwortrichtlinie für benutzerdefinierte Konten über die AWS API zu verwalten

Rufen Sie die folgenden Operationen auf:

- So erstellen oder ändern Sie die Richtlinie für benutzerdefinierte Passwörter: `UpdateAccountPasswordPolicy`
- So zeigen Sie die Passwortrichtlinie an: `GetAccountPasswordPolicy`
- So löschen Sie die Richtlinie für benutzerdefinierte Passwörter: `DeleteAccountPasswordPolicy`

Verwalten von Passwörtern für IAM-Benutzer

IAM-Benutzer, die den AWS Management Console für die Arbeit mit AWS Ressourcen verwenden, benötigen ein Passwort, um sich anmelden zu können. Sie können ein Passwort für einen IAM-Benutzer in Ihrem AWS -Konto erstellen, ändern oder löschen.

Nachdem Sie einem Benutzer ein Passwort zugewiesen haben, kann sich der Benutzer AWS Management Console mit der Anmelde-URL für Ihr Konto anmelden, die wie folgt aussieht:

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Weitere Informationen darüber, wie sich IAM-Benutzer bei der anmelden AWS Management Console, finden Sie unter So melden Sie sich an AWS im AWS-Anmeldung Benutzerhandbuch.

Auch wenn Ihre Benutzer ihre eigenen Passwörter haben, benötigen Sie dennoch Berechtigungen für den Zugriff auf Ihre AWS -Ressourcen. Standardmäßig hat ein Benutzer keine Berechtigungen. Um Ihren Benutzern die erforderlichen Berechtigungen zu gewähren, weisen Sie ihnen oder den Gruppen, zu denen sie gehören, Richtlinien zu. Weitere Informationen zum Erstellen von Benutzern und Gruppen finden Sie unter IAM-Identitäten (Benutzer, Gruppen und Rollen). Weitere Informationen zum Verwenden von Richtlinien, um Berechtigungen festzulegen, finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer.

Sie können Benutzern die Berechtigung zuweisen, ihre eigenen Passwörter zu ändern. Weitere Informationen finden Sie unter Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können. Weitere Informationen zum Zugriff auf die Anmeldeseite durch Benutzer finden Sie unter Anmelden bei AWS im AWS-Anmeldung -Benutzerhandbuch.

Themen

- Erstellen, Ändern oder Löschen eines IAM-Benutzerpassworts (Konsole)
- Erstellen, Ändern oder Löschen eines IAM-Benutzerpassworts (AWS CLI)
- Ein IAM-Benutzerkennwort (API) erstellen, ändern oder löschen AWS

Erstellen, Ändern oder Löschen eines IAM-Benutzerpassworts (Konsole)

Sie können den verwenden AWS Management Console , um Passwörter für Ihre IAM-Benutzer zu verwalten.

Wenn Benutzer Ihr Unternehmen verlassen oder keinen AWS Zugriff mehr benötigen, ist es wichtig, die von ihnen verwendeten Anmeldeinformationen zu finden und sicherzustellen, dass sie nicht mehr

betriebsbereit sind. Im Idealfall löschen Sie die Anmeldeinformationen, wenn sie nicht mehr benötigt werden. Sie können sie immer zu einem späteren Zeitpunkt bei Bedarf neu erstellen. Zumindest sollten Sie die Anmeldeinformationen so ändern, dass die ehemaligen Benutzer keinen Zugriff mehr haben.

So fügen Sie ein Passwort für einen IAM-Benutzer hinzu (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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Passwort Sie erstellen möchten.
4. Wählen Sie die Registerkarte Security credentials (Sicherheitsanmeldeinformationen) und wählen Sie dann unter Console sign-in (Konsolenanmeldung) die Option Enable console access (Konsolenzugriff aktivieren) aus.
5. Wählen Sie unter Konsolenzugriff aktivieren für das Konsolenpasswort aus, ob IAM ein Passwort generieren oder ein benutzerdefiniertes Passwort erstellen soll:

   - Wenn IAM ein Passwort generieren soll, wählen Sie Automatisch generiertes Passwort.
   - Wenn ein benutzerdefiniertes Passwort erstellt werden soll, wählen Sie Custom password (Benutzerdefiniertes Passwort) aus und geben Sie das Passwort ein.

   > **Note**
   >
   > Das von Ihnen erstellte Passwort muss den Passwortrichtlinie des Kontos entsprechen.

6. Wenn der Benutzer bei der Anmeldung ein neues Passwort erstellen soll, wählen Sie User must create a new password at next sign-in (Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen). Wählen Sie dann Konsolenzugriff aktivieren aus.

   > **Important**
   >
   > Wenn Sie die Option User must create a new password at next sign-in (Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen) auswählen, vergewissern Sie sich, dass der Benutzer die Berechtigung zum Ändern des Passworts hat. Weitere Informationen finden Sie unter Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können.

7. Um das Passwort anzuzeigen, sodass Sie es mit dem Benutzer teilen können, wählen Sie im Dialogfeld „Konsolenkennwort“ die Option „Anzeigen“.

> **Important**
>
> Aus Sicherheitsgründen können Sie nach Ausführen dieses Schritts nicht auf das Passwort zugreifen, Sie können jedoch jederzeit ein neues Passwort erstellen.

So ändern Sie das Passwort für einen IAM-Benutzer (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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Passwort Sie ändern möchten.
4. Wählen Sie die Registerkarte Security credentials (Sicherheitsanmeldeinformationen) und wählen Sie dann unter Console sign-in (Konsolenanmeldung) die Option Manage console access (Konsolenzugriff verwalten) aus.
5. Wählen Sie unter Konsolenzugriff verwalten die Option Passwort zurücksetzen aus, falls diese Option noch nicht ausgewählt ist. Wenn der Konsolenzugriff deaktiviert ist, ist kein Passwort erforderlich.
6. Wählen Sie für den Konsolenzugriff, ob IAM ein Passwort generieren oder ein benutzerdefiniertes Passwort erstellen soll:
   - Wenn IAM ein Passwort generieren soll, wählen Sie Automatisch generiertes Passwort.
   - Wenn ein benutzerdefiniertes Passwort erstellt werden soll, wählen Sie Custom password (Benutzerdefiniertes Passwort) aus und geben Sie das Passwort ein.

   > **Note**
   >
   > Das von Ihnen erstellte Passwort muss der Passwortrichtlinie des Kontos entsprechen, sofern zurzeit eine festgelegt ist.
7. Wenn der Benutzer bei der Anmeldung ein neues Passwort erstellen soll, wählen Sie User must create a new password at next sign-in (Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen).

> **Important**
>
> Wenn Sie die Option User must create a new password at next sign-in (Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen) auswählen, vergewissern Sie sich, dass der Benutzer die Berechtigung zum Ändern des Passworts hat. Weitere Informationen finden Sie unter Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können.

8. Um die aktiven Konsolensitzungen des Benutzers zu widerrufen, wählen Sie Aktive Konsolensitzungen widerrufen. Wählen Sie dann Apply (Anwenden).

   Wenn Sie aktive Konsolensitzungen für einen Benutzer widerrufen, fügt IAM dem Benutzer eine neue Inline-Richtlinie hinzu, die ihm alle Berechtigungen für alle Aktionen verweigert. Es beinhaltet eine Bedingung, die die Einschränkungen nur dann anwendet, wenn die Sitzung vor dem Zeitpunkt, zu dem Sie die Berechtigungen widerrufen, erstellt wurde, sowie etwa 30 Sekunden in der future. Wenn der Benutzer eine neue Sitzung erstellt, nachdem Sie die Berechtigungen widerrufen haben, gilt die Ablehnungsrichtlinie nicht für diesen Benutzer. Wenn ein Benutzer seine eigenen aktiven Konsolensitzungen mit dieser Methode widerruft, wird er sofort von der abgemeldet. AWS Management Console

   > **Important**
   >
   > Um aktive Konsolensitzungen für einen Benutzer erfolgreich zu widerrufen, benötigen Sie die `PutUserPolicy` entsprechende Benutzerberechtigung. Auf diese Weise können Sie die `AWSRevokeOlderSessions` Inline-Richtlinie an den Benutzer anhängen.

9. Um das Passwort anzuzeigen, sodass Sie es mit dem Benutzer teilen können, wählen Sie im Dialogfeld „Konsolenkennwort“ die Option „Anzeigen“.

   > **Important**
   >
   > Aus Sicherheitsgründen können Sie nach Ausführen dieses Schritts nicht auf das Passwort zugreifen, Sie können jedoch jederzeit ein neues Passwort erstellen.

So löschen (deaktivieren) Sie das Passwort eines IAM-Benutzers (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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers aus, dessen Passwort Sie löschen möchten.
4. Wählen Sie die Registerkarte Security credentials (Sicherheitsanmeldeinformationen) und wählen Sie dann unter Console sign-in (Konsolenanmeldung) die Option Manage console access (Konsolenzugriff verwalten) aus.
5. Wählen Sie unter Konsolenzugriff verwalten die Option Konsolenzugriff deaktivieren aus, falls diese Option noch nicht ausgewählt ist. Wenn der Konsolenzugriff deaktiviert ist, ist kein Passwort erforderlich.
6. Um die aktiven Konsolensitzungen des Benutzers zu widerrufen, wählen Sie Aktive Konsolensitzungen widerrufen. Wählen Sie dann Zugriff deaktivieren.

   > **Important**
   >
   > Um aktive Konsolensitzungen für einen Benutzer erfolgreich zu widerrufen, benötigen Sie die `PutUserPolicy` entsprechende Benutzerberechtigung. Auf diese Weise können Sie die `AWSRevokeOlderSessions` Inline-Richtlinie an den Benutzer anhängen.

   Wenn Sie aktive Konsolensitzungen für einen Benutzer widerrufen, bettet IAM eine neue Inline-Richtlinie in den IAM-Benutzer ein, die alle Berechtigungen für alle Aktionen verweigert. Es beinhaltet eine Bedingung, die die Einschränkungen nur dann anwendet, wenn die Sitzung vor dem Zeitpunkt, zu dem Sie die Berechtigungen widerrufen, erstellt wurde, sowie etwa 30 Sekunden in der future. Wenn der Benutzer eine neue Sitzung erstellt, nachdem Sie die Berechtigungen widerrufen haben, gilt die Ablehnungsrichtlinie nicht für diesen Benutzer. Wenn ein Benutzer seine eigenen aktiven Konsolensitzungen mit dieser Methode widerruft, wird er sofort von der abgemeldet. AWS Management Console

> **Important**
>
> Sie können verhindern, dass ein IAM-Benutzer auf die zugreift, AWS Management Console indem Sie sein Passwort entfernen. Dadurch wird verhindert, dass sie sich AWS Management Console mit ihren Anmeldeinformationen bei der anmelden. Es ändert

weder ihre Berechtigungen noch verhindert, dass sie mit einer angenommenen Rolle auf die Konsole zugreifen. Wenn der Benutzer über aktive Zugriffsschlüssel verfügt, funktionieren diese weiterhin und ermöglichen den Zugriff über die AWS CLI Tools für Windows PowerShell, die AWS API oder die AWS Console Mobile Application.

Erstellen, Ändern oder Löschen eines IAM-Benutzerpassworts (AWS CLI)

Sie können die AWS CLI API verwenden, um Passwörter für Ihre IAM-Benutzer zu verwalten.

So erstellen Sie ein Passwort (AWS CLI)

1. (Optional) Um festzustellen, ob ein Benutzer ein Passwort hat, führen Sie diesen Befehl aus: aws iam get-login-profile
2. Führen Sie diesen Befehl aus, um ein Passwort zu erstellen: aws iam create-login-profile

So ändern Sie das Passwort eines Benutzers (AWS CLI)

1. (Optional) Um festzustellen, ob ein Benutzer ein Passwort hat, führen Sie diesen Befehl aus: aws iam get-login-profile
2. Um ein Passwort zu ändern, führen Sie diesen Befehl aus: aws iam update-login-profile

So löschen (deaktivieren) Sie das Passwort eines Benutzers (AWS CLI)

1. (Optional) Um festzustellen, ob ein Benutzer ein Passwort hat, führen Sie diesen Befehl aus: aws iam get-login-profile
2. (Optional) Um zu ermitteln, wann ein Passwort zuletzt verwendet wurde, führen Sie diesen Befehl aus: aws iam get-user
3. Um ein Passwort zu löschen, führen Sie diesen Befehl aus: aws iam delete-login-profile

Important

Wenn Sie das Passwort eines Benutzers löschen, kann sich dieser nicht mehr bei der AWS Management Console anmelden. Wenn der Benutzer über aktive Zugriffsschlüssel verfügt, funktionieren diese weiterhin und ermöglichen den Zugriff über die AWS CLI Funktionsaufrufen Tools für Windows PowerShell oder AWS API. Wenn Sie die AWS CLI Tools für Windows oder die AWS API verwenden PowerShell, um einen Benutzer aus Ihrem

zu löschen AWS-Konto, müssen Sie zuerst das Passwort mithilfe dieses Vorgangs löschen. Weitere Informationen finden Sie unter Löschen eines IAM-Benutzers (AWS CLI).

Um die aktiven Konsolensitzungen eines Benutzers vor einer bestimmten Zeit zu widerrufen (AWS CLI)

1. Um eine Inline-Richtlinie einzubetten, die die aktiven Konsolensitzungen eines IAM-Benutzers vor einem bestimmten Zeitpunkt widerruft, verwenden Sie die folgende Inline-Richtlinie und führen Sie diesen Befehl aus: aws iam put-user-policy

   Diese Inline-Richtlinie verweigert alle Berechtigungen und beinhaltet den Bedingungsschlüssel. `aws: TokenIssue Zeit` Sie widerruft die aktiven Konsolensitzungen des Benutzers vor dem in der `Condition` Inline-Richtlinie angegebenen Zeitpunkt. Ersetzen Sie den Wert des `aws:TokenIssueTime` Bedingungsschlüssels durch Ihren eigenen Wert.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "*",
       "Resource": "*",
       "Condition": {
         "DateLessThan": {
           "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
         }
       }
     }
   }
   ```

2. (Optional) Um die Namen der im IAM-Benutzer eingebetteten Inline-Richtlinien aufzulisten, führen Sie diesen Befehl aus: aws iam list-user-policies
3. (Optional) Um die benannte Inline-Richtlinie anzuzeigen, die in den IAM-Benutzer eingebettet ist, führen Sie diesen Befehl aus: aws iam get-user-policy

Ein IAM-Benutzerkennwort (API) erstellen, ändern oder löschen AWS

Sie können die AWS API verwenden, um Passwörter für Ihre IAM-Benutzer zu verwalten.

Um ein Passwort (AWS API) zu erstellen

1. (Optional) Rufen Sie diesen Vorgang auf, um festzustellen, ob ein Benutzer ein Passwort hat: GetLoginProfile
2. Rufen Sie diesen Vorgang auf, um ein Passwort zu erstellen: CreateLoginProfile

Um das Passwort eines Benutzers zu ändern (AWS API)

1. (Optional) Rufen Sie diesen Vorgang auf, um festzustellen, ob ein Benutzer ein Passwort hat: GetLoginProfile
2. Rufen Sie diesen Vorgang auf, um ein Passwort zu ändern: UpdateLoginProfile

Um das Passwort (AWS API) eines Benutzers zu löschen (zu deaktivieren)

1. (Optional) Um festzustellen, ob ein Benutzer ein Passwort hat, führen Sie diesen Befehl aus: GetLoginProfile
2. (Optional) Um festzustellen, wann ein Passwort zuletzt verwendet wurde, führen Sie diesen Befehl aus: GetUser
3. Führen Sie diesen Befehl aus, um ein Passwort zu löschen: DeleteLoginProfile

> **Important**
>
> Wenn Sie das Passwort eines Benutzers löschen, kann sich dieser nicht mehr bei der AWS Management Console anmelden. Wenn der Benutzer über aktive Zugriffsschlüssel verfügt, funktionieren diese weiterhin und ermöglichen den Zugriff über die AWS CLI Funktionsaufrufen Tools für Windows PowerShell oder AWS API. Wenn Sie die AWS CLI Tools für Windows oder die AWS API verwenden PowerShell, um einen Benutzer aus Ihrem zu löschen AWS-Konto, müssen Sie zuerst das Passwort mithilfe dieses Vorgangs löschen. Weitere Informationen finden Sie unter Löschen eines IAM-Benutzers (AWS CLI).

Um die aktiven Konsolensitzungen eines Benutzers vor einem bestimmten Zeitpunkt zu widerrufen (AWS API)

1. Um eine Inline-Richtlinie einzubetten, die die aktiven Konsolensitzungen eines IAM-Benutzers vor einem bestimmten Zeitpunkt widerruft, verwenden Sie die folgende Inline-Richtlinie und führen Sie diesen Befehl aus: PutUserPolicy

   Diese Inline-Richtlinie verweigert alle Berechtigungen und beinhaltet den `aws: TokenIssue Zeit` Bedingungsschlüssel. Sie widerruft die aktiven Konsolensitzungen des Benutzers vor dem in der `Condition` Inline-Richtlinie angegebenen Zeitpunkt. Ersetzen Sie den Wert des `aws:TokenIssueTime` Bedingungsschlüssels durch Ihren eigenen Wert.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "*",
       "Resource": "*",
       "Condition": {
         "DateLessThan": {
           "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
         }
       }
     }
   }
   ```

2. (Optional) Um die Namen der Inline-Richtlinien aufzulisten, die in den IAM-Benutzer eingebettet sind, führen Sie diesen Befehl aus: ListUserPolicies
3. (Optional) Führen Sie den folgenden Befehl aus, um die benannte Inline-Richtlinie anzuzeigen, die in den IAM-Benutzer eingebettet ist: GetUserPolicy

Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können

> **Note**
>
> Benutzer mit Verbundidentitäten verwenden den von ihrem Identitätsanbieter definierten Prozess, um ihre Passwörter zu ändern. Als bewährte Methode sollten menschliche Benutzer für den Zugriff AWS mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden.

Sie können IAM-Benutzern die Berechtigung gewähren, ihr eigenes Passwort für die Anmeldung bei der AWS Management Console zu ändern. Dafür stehen Ihnen zwei Optionen zur Verfügung:

- Gewähren Sie allen IAM-Benutzern im Konto die Berechtigung, ihr eigenes Passwort zu ändern.
- Gewähren Sie nur ausgewählten IAM-Benutzern, ihr eigenes Passwort zu ändern. In diesem Szenario deaktivieren Sie die Option für alle Benutzer, ihre eigenen Passwörter zu ändern, und Sie verwenden eine IAM-Richtlinie, um nur einigen Benutzern Berechtigungen zu gewähren. Dieser Ansatz ermöglicht es diesen Benutzern, ihre eigenen Passwörter und optional andere Anmeldeinformationen wie ihre eigenen Zugriffsschlüssel zu ändern.

> **Important**
>
> Wir empfehlen, dass Sie eine benutzerdefinierte Passwortrichtlinie festlegen, nach der IAM-Benutzer sichere Passwörter erstellen müssen.

So ermöglichen Sie allen IAM-Benutzern, ihr eigenes Passwort zu ändern

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 auf Kontoeinstellungen.
3. Wählen Sie im Abschnitt Password policy (Passwortrichtlinie) die Option Edit (Bearbeiten) aus.
4. Wählen Sie Custom (Benutzerdefiniert), um eine benutzerdefinierte Passwortrichtlinie zu verwenden.
5. Wählen Sie Allow users to change their own password (Benutzern erlauben, ihr eigenes Passwort zu ändern), und dann Save changes (Änderungen speichern). Dadurch haben alle Benutzer im Konto Zugriff auf die `iam:ChangePassword`-Aktion nur für ihren Benutzer und auf die `iam:GetAccountPasswordPolicy`-Aktion.
6. Geben Sie Benutzern die folgenden Anweisungen zum Ändern ihrer Passwörter: Wie ein IAM-Benutzer sein eigenes Passwort ändert.

Informationen zu den Befehlen AWS CLI, Tools für Windows und API-Befehle PowerShell, mit denen Sie die Passwortrichtlinie des Kontos ändern können (einschließlich der Möglichkeit, dass alle Benutzer ihre eigenen Passwörter ändern können), finden Sie unterEinrichten einer Passwortrichtlinie (AWS CLI).

So ermöglichen Sie es ausgewählten IAM-Benutzern, ihr eigenes Passwort zu ändern

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 auf Kontoeinstellungen.
3. Stellen Sie im Bereich Kontoeinstellungen sicher, dass die Option Benutzern erlauben, ihr eigenes Passwort zu ändern deaktiviert ist. Wenn dieses Kontrollkästchen aktiviert ist, können alle Benutzer ihr eigenes Passwort ändern (siehe vorheriger Abschnitt).
4. Erstellen Sie gegebenenfalls die Benutzer, die ihr eigenes Passwort ändern dürfen. Details hierzu finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto.
5. (Optional) Erstellen Sie eine IAM-Gruppe für die Benutzer, die ihr Passwort ändern dürfen, und fügen Sie die Benutzer aus dem vorherigen Schritt der Gruppe hinzu. Details hierzu finden Sie unter Verwalten von IAM-Gruppen.
6. Weisen Sie der Gruppe die folgende Richtlinie zu. Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Diese Richtlinie gewährt Zugriff auf die ChangePasswordAktion, mit der Benutzer nur ihre eigenen Passwörter über die Konsole AWS CLI, die Tools für Windows PowerShell oder die API ändern können. Sie gewährt auch Zugriff auf die GetAccountPasswordPolicyAktion, mit der der Benutzer die aktuelle Kennwortrichtlinie einsehen kann. Diese Berechtigung ist erforderlich, damit der Benutzer die Kontokennwortrichtlinie auf der Seite Passwort ändern einsehen kann.

Benutzer müssen die aktuelle Passwortrichtlinie lesen können, um sicherzustellen, dass das neue Passwort den Anforderungen der Richtlinie entspricht.

7. Geben Sie Benutzern die folgenden Anweisungen zum Ändern ihrer Passwörter: Wie ein IAM-Benutzer sein eigenes Passwort ändert.

Weitere Informationen

Weitere Informationen zum Verwalten von Anmeldeinformationen finden Sie in den folgenden Themen:

- Zulassen, dass IAM-Benutzer ihr eigenes Passwort ändern können
- Benutzerpasswörter verwalten in AWS
- Einrichten einer Kontopasswortrichtlinie für IAM-Benutzer
- Verwalten von IAM-Richtlinien
- Wie ein IAM-Benutzer sein eigenes Passwort ändert

Wie ein IAM-Benutzer sein eigenes Passwort ändert

Wenn Ihnen die Erlaubnis erteilt wurde, Ihr eigenes IAM-Benutzerkennwort zu ändern, können Sie AWS Management Console dazu eine spezielle Seite in der verwenden. Sie können auch die AWS API AWS CLI oder verwenden.

Themen

- Erforderliche Berechtigungen
- Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole)
- Wie ändern IAM-Benutzer ihr eigenes Passwort (AWS CLI oder ihre AWS API)

Erforderliche Berechtigungen

Um das Passwort für Ihren eigenen IAM-Benutzer zu ändern, benötigen Sie die Berechtigungen von der folgenden Richtlinie: AWS: Ermöglicht IAM-Benutzern, ihr eigenes Konsolenkennwort auf der Seite mit den Sicherheitsanmeldedaten zu ändern.

Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole)

Das folgende Verfahren beschreibt, wie IAM-Benutzer das verwenden können AWS Management Console , um ihr eigenes Passwort zu ändern.

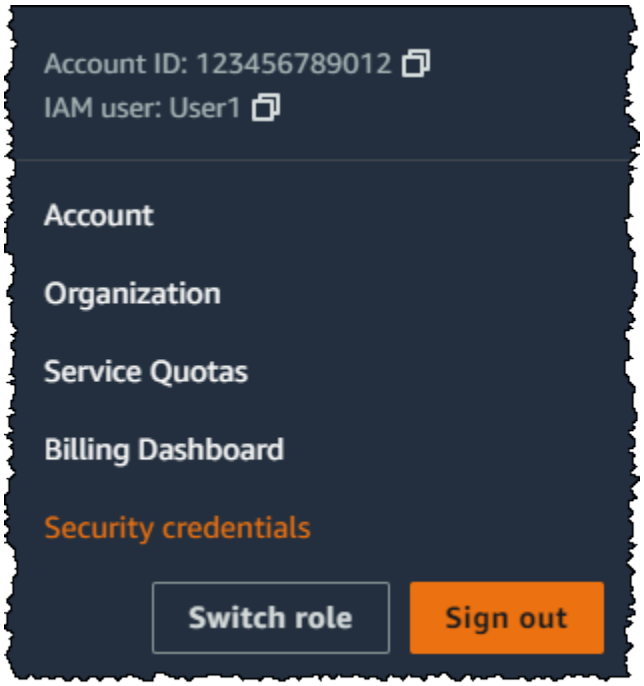
So ändern Sie Ihr eigenes IAM-Benutzer-Passwort (Konsole)

1. Verwenden Sie Ihre AWS Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der IAM-Konsole anzumelden.

   > **Note**
   >
   > Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich Ihren IAM-Benutzernamen und Ihre Kontoinformationen zu merken. Wenn Sie sich zuvor als anderer Benutzer angemeldet haben, wählen Sie Melden Sie sich bei einem anderen Konto an Um zur Hauptanmeldeseite zurückzukehren. Von dort aus können Sie Ihre AWS Konto-ID oder Ihren Kontoalias eingeben, um zur IAM-Benutzer-Anmeldeseite für Ihr Konto weitergeleitet zu werden.

   Wenden Sie sich an Ihren Administrator, um Ihre AWS-Konto ID zu erhalten.

2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheitsanmeldeinformationen) aus.



3. Wählen Sie auf der Registerkarte AWS -IAM-Anmeldeinformationen die Option Passwort aktualisieren aus.

4. Geben Sie unter Current Password (Aktuelles Passwort) Ihr aktuelles Passwort ein. Geben Sie in den Feldern New Passwort (Neues Passwort) und Confirm new password (Neues Passwort bestätigen) ein neues Passwort ein. Klicken Sie dann auf Passwort ändern.

> **Note**
>
> Wenn das Konto über eine Passwortrichtlinie verfügt, muss das neue Passwort die Anforderungen dieser Richtlinie erfüllen. Weitere Informationen finden Sie unter Einrichten einer Kontopasswortrichtlinie für IAM-Benutzer.

Wie ändern IAM-Benutzer ihr eigenes Passwort (AWS CLI oder ihre AWS API)

Das folgende Verfahren beschreibt, wie IAM-Benutzer die AWS CLI AWS OR-API verwenden können, um ihr eigenes Passwort zu ändern.

Ändern Sie Ihr IAM-Passwort wie folgt:

- AWS CLI: `aws iam change-password`
- AWS API: `ChangePassword`

Verwalten der Zugriffsschlüssel für IAM-Benutzer

Follow us on Twitter

> **Important**
>
> Als bewährte Methode empfiehlt es sich, temporäre Sicherheitsanmeldeinformationen (z. B. IAM-Rollen) zu verwenden, anstatt langfristige Anmeldeinformationen wie Zugriffsschlüssel zu erstellen. Bevor Sie Zugriffsschlüssel erstellen, prüfen Sie die Alternativen zu Langzeit-Zugriffsschlüsseln.

Zugriffsschlüssel sind langfristige Anmeldeinformationen für einen IAM-Benutzer oder Root-Benutzer des AWS-Kontos. Sie können Zugriffstasten verwenden, um programmatische Anfragen an die AWS API AWS CLI oder zu signieren (direkt oder mithilfe des AWS SDK). Weitere Informationen finden Sie unter AWS API-Anfragen signieren.

Zugriffsschlüssel bestehen aus zwei Teilen: einer Zugriffsschlüssel-ID (z. B. `AKIAIOSFODNN7EXAMPLE`) und einem geheimen Zugriffsschlüssel (z. B. `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Sie müssen die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel zusammen verwenden, um Ihre Anforderungen zu authentifizieren.

Speichern Sie beim Erstellen eines Zugriffsschlüsselpaars die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel an einem sicheren Speicherort. Der geheime Zugriffsschlüssel ist nur zu dem Zeitpunkt verfügbar, an dem Sie ihn erstellen. Wenn Sie Ihren geheimen Zugriffsschlüssel verlieren, müssen Sie den Zugriffsschlüssel löschen und einen neuen erstellen. Weitere Details finden Sie unter Zurücksetzen verlorener oder vergessener Passwörter oder Zugangsschlüssel für AWS.

Sie können maximal zwei Zugriffsschlüssel pro Benutzer besitzen.

> **Important**
>
> Verwalten Sie Ihre Zugriffsschlüssel auf sichere Weise. Geben Sie Ihre Zugangsschlüssel nicht an Unbefugte weiter, auch nicht, um Ihre Kontokennungen zu finden. Wenn Sie dies tun, gewähren Sie anderen Personen möglicherweise den permanenten Zugriff auf Ihr Konto.

In den folgenden Themen werden die Verwaltungsaufgaben im Zusammenhang mit Zugriffsschlüsseln detailliert beschrieben.

Themen

Erforderliche Berechtigungen zum Verwalten von Zugriffsschlüsseln

> **Note**
>
> `iam:TagUser` ist eine optionale Berechtigung zum Hinzufügen und Bearbeiten von Beschreibungen für den Zugriffsschlüssel. Weitere Informationen finden Sie unter Markieren von IAM-Benutzern.

Um Zugriffsschlüssel für Ihren eigenen IAM-Benutzer zu erstellen, benötigen Sie die Berechtigungen von der folgenden Richtlinie:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Um Zugriffsschlüssel für Ihren eigenen IAM-Benutzer zu aktualisieren, müssen Sie über die Berechtigungen der folgenden Richtlinie verfügen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
```

```
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Verwalten von Zugriffsschlüsseln (Konsole)

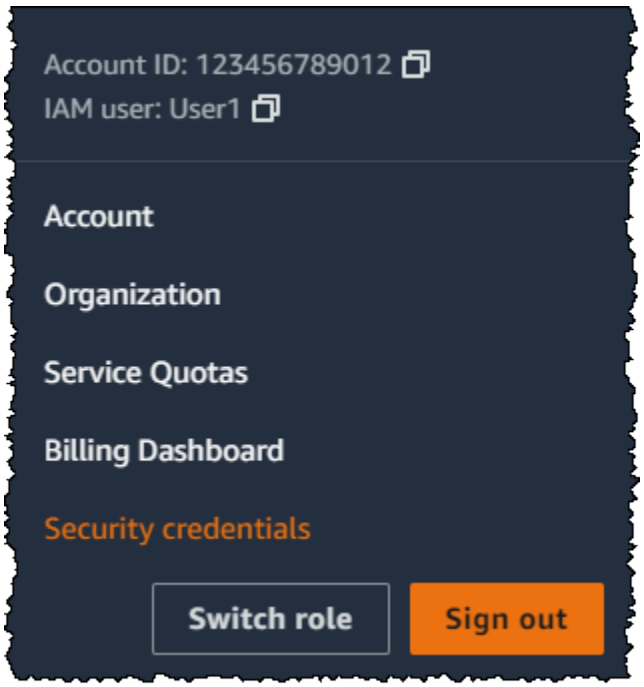
Sie können den verwenden AWS Management Console , um die Zugriffsschlüssel eines IAM-Benutzers zu verwalten.

So erstellen, ändern oder löschen Sie Ihre eigenen Zugriffsschlüssel (Konsole)

1. Verwenden Sie Ihre AWS Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der IAM-Konsole anzumelden.

   > **Note**
   >
   > Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich Ihren IAM-Benutzernamen und Ihre Kontoinformationen zu merken. Wenn Sie sich zuvor als anderer Benutzer angemeldet haben, wählen Sie Melden Sie sich bei einem anderen Konto an Um zur Hauptanmeldeseite zurückzukehren. Von dort aus können Sie Ihre AWS Konto-ID oder Ihren Kontoalias eingeben, um zur IAM-Benutzer-Anmeldeseite für Ihr Konto weitergeleitet zu werden.

   Wenden Sie sich an Ihren Administrator, um Ihre AWS-Konto ID zu erhalten.
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheitsanmeldeinformationen) aus.



Führen Sie eine der folgenden Aktionen aus:

So erstellen Sie einen Zugriffsschlüssel

1. Wählen Sie im Abschnitt Access keys (Zugriffsschlüssel) Create access key (Zugriffsschlüssel erstellen). Wenn Sie bereits über zwei Zugriffsschlüssel verfügen, ist diese Schaltfläche deaktiviert und Sie müssen einen Zugriffsschlüssel löschen, bevor Sie einen neuen erstellen können.
2. Wählen Sie auf der Seite Access key best practices & alternatives (Bewährte Methoden und Alternativen zu Zugriffsschlüsseln) Ihren Anwendungsfall aus, um mehr über zusätzliche Optionen zu erfahren, mit denen Sie die Erstellung eines langfristigen Zugriffsschlüssels vermeiden können. Wenn Sie feststellen, dass für Ihren Anwendungsfall immer noch ein Zugriffsschlüssel erforderlich ist, wählen Sie Other (Andere) und dann Next (Weiter).
3. (Optional) Legen Sie einen Beschreibungs-Tag-Wert für den Zugriffsschlüssel fest. Dadurch wird Ihrem IAM-Benutzer ein Tag-Schlüssel/Wert-Paar hinzugefügt. Auf diese Weise können Sie Zugriffsschlüssel leichter identifizieren und aktualisieren. Der Tag-Schlüssel ist auf die Zugriffsschlüssel-ID festgelegt. Der Tag-Wert ist auf die von Ihnen angegebene Beschreibung des Zugriffsschlüssels festgelegt. Wenn Sie fertig sind, wählen Sie Create access key (Zugriffsschlüssel erstellen) aus.

4. Wählen Sie auf der Seite Retrieve access keys (Zugriffsschlüssel abrufen) entweder Show (Anzeigen), um den Wert des geheimen Zugriffsschlüssels Ihres Benutzers anzuzeigen, oder Download .csv file (CSV-Datei herunterladen). Das ist Ihre einzige Gelegenheit, Ihren geheimen Zugriffsschlüssel zu speichern. Nachdem Sie Ihren geheimen Zugriffsschlüssel an einem sicheren Ort gespeichert haben, wählen Sie Done (Fertig) aus.

So deaktivieren Sie einen Zugriffsschlüssel

- Suchen Sie im Abschnitt Access keys (Zugriffsschlüssel) nach dem Schlüssel, den Sie deaktivieren möchten, und wählen Sie dann Actions (Aktionen) und anschließend Deactivate (Deaktivieren). Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Deaktivieren aus. Ein deaktivierter Zugriffsschlüssel wird noch immer auf das Limit von zwei Zugriffsschlüsseln angerechnet.

So aktivieren Sie einen Zugriffsschlüssel

- Suchen Sie im Abschnitt Access keys (Zugriffsschlüssel) nach dem Schlüssel, den Sie aktivieren möchten, und wählen Sie dann Actions (Aktionen) und anschließend Activate (Aktivieren).

So löschen Sie einen Zugriffsschlüssel, den Sie nicht mehr benötigen

- Suchen Sie im Abschnitt Access keys (Zugriffsschlüssel) nach dem Schlüssel, den Sie löschen möchten, und wählen Sie dann Actions (Aktionen) und anschließend Delete (Löschen). Folgen Sie den Anweisungen im Dialogfeld, um zuerst zu Deactivate (Deaktivieren) und dann den Löschvorgang zu bestätigen. Sie sollten sich vergewissern, dass der Zugriffsschlüssel wirklich nicht mehr verwendet wird, bevor Sie ihn endgültig löschen.

So erstellen, ändern oder löschen Sie die Zugriffsschlüssel eines anderen IAM-Benutzers (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 auf Users (Benutzer).
3. Wählen Sie erst den Namen des Benutzers, dessen Zugriffsschlüssel Sie verwalten möchten, und dann die Registerkarte Security credentials (Sicherheitsanmeldeinformationen).
4. Führen Sie im Bereich Access keys (Zugriffsschlüssel) einen der folgenden Schritte aus:

- Um einen Zugriffsschlüssel zu erstellen, wählen Sie Create access key (Zugriffsschlüssel erstellen). Wenn diese Schaltfläche deaktiviert ist, müssen Sie einen der vorhandenen Schlüssel löschen, bevor Sie einen neuen erstellen können. Sehen Sie sich auf der Seite Access key best practices & alternatives (Bewährte Methoden und Alternativen zu Zugriffsschlüsseln) die wichtigsten bewährten Methoden und Alternativen an. Wählen Sie Ihren Anwendungsfall aus, um mehr über zusätzliche Optionen zu erfahren, mit denen Sie die Erstellung eines langfristigen Zugriffsschlüssels vermeiden können. Wenn Sie feststellen, dass für Ihren Anwendungsfall immer noch ein Zugriffsschlüssel erforderlich ist, wählen Sie Other (Andere) und dann Next (Weiter). Wählen Sie auf der Seite Retrieve access keys (Zugriffsschlüssel abrufen) Show (Anzeigen), um den Wert des geheimen Zugriffsschlüssels Ihres Benutzers anzuzeigen. Um die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel in einer `.csv`-Dateien einem sicheren Ort auf Ihrem Computer zu speichern, wählen Sie die Schaltfläche Download .csv file (CSV-Datei herunterladen). Wenn Sie einen Zugriffsschlüssel erstellen, ist das Schlüsselpaar standardmäßig aktiv, und Ihr Benutzer kann es sofort verwenden.
- Um einen aktiven Zugriffsschlüssel zu deaktivieren, wählen Sie Actions (Aktionen) und dann Deactivate (Deaktivieren).
- Um einen inaktiven Zugriffsschlüssel zu aktivieren, wählen Sie Actions (Aktionen) und dann Activate (Aktivieren).
- Um Ihren Zugriffsschlüssel zu löschen, wählen Sie Actions (Aktionen) und dann Delete (Löschen). Folgen Sie den Anweisungen im Dialog, um zuerst zu deaktivieren und dann den Löschvorgang zu bestätigen. AWS empfiehlt, vorher den Schlüssel zu deaktivieren und zu testen, ob er nicht mehr verwendet wird. Wenn Sie den verwenden AWS Management Console, müssen Sie Ihren Schlüssel deaktivieren, bevor Sie ihn löschen können.

So listen Sie die Zugriffsschlüssel für einen IAM-Benutzer auf (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 auf Users (Benutzer).
3. Wählen Sie den Namen des gewünschten Benutzers und dann die Registerkarte Security credentials (Sicherheitsanmeldeinformationen). Im Abschnitt Access keys (Zugriffsschlüssel) werden die Zugriffsschlüssel des Benutzers und der Status der einzelnen Schlüssel angezeigt.

> **Note**
>
> Nur die Zugriffsschlüssel-ID des Benutzers wird angezeigt. Der geheime Zugriffsschlüssel kann nur während der Erstellung des Schlüssels abgerufen werden.

So listen Sie die Zugriffsschlüssel-IDs für mehrere IAM-Benutzer auf (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 auf Users (Benutzer).
3. Falls erforderlich, fügen Sie die Spalte Access key ID (Zugriffsschlüssel-ID) zur Tabelle der Benutzer hinzu, indem Sie die folgenden Schritte ausführen:
   a. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ( ).
   b. Wählen Sie in der Spalte Manage Columns (Spalten verwalten) die Option Access key ID (Zugriffsschlüssel-ID).
   c. Klicken Sie auf Close (Schließen), um zur Liste der Benutzer zurückzukehren.
4. Die Spalte Access key ID (Zugriffsschlüssel-ID) enthält alle Zugriffsschlüssel-IDs, gefolgt vom Status, z. B. 2347820702784207323076237 4023 (Active) (2347820702784207323076237 4023 (Aktiv)) oder 2209374023967023702484342 0327 (Inactive) (2209374023967023702484342 0327 (Inaktiv)).

   Anhand dieser Informationen können Sie die Zugriffsschlüssel für Benutzer mit einem oder zwei Zugriffsschlüsseln anzeigen und kopieren. Für Benutzer ohne Zugriffsschlüssel wird in der Spalte None (Keiner) angezeigt.

> **Note**
>
> Nur die Zugriffsschlüssel-ID des Benutzers und ihr Status werden angezeigt. Der geheime Zugriffsschlüssel kann nur während der Erstellung des Schlüssels abgerufen werden.

So ermitteln Sie IAM-Benutzer mit einem bestimmten Zugriffsschlüssel (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 auf Users (Benutzer).
3. Kopieren oder geben Sie in das Suchfeld die Zugriffsschlüssel-ID des Benutzers ein, den Sie suchen möchten.
4. Falls erforderlich, fügen Sie die Spalte Access key ID (Zugriffsschlüssel-ID) zur Tabelle der Benutzer hinzu, indem Sie die folgenden Schritte ausführen:
   a. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ().
   b. Wählen Sie in der Spalte Manage Columns (Spalten verwalten) die Option Access key ID (Zugriffsschlüssel-ID).
   c. Klicken Sie auf Close (Schließen), um zur Liste der Benutzer zurückzukehren, und vergewissern Sie sich, dass der gefilterte Benutzer Eigentümer des angegebenen Zugriffsschlüssels ist.

Verwalten von Zugriffsschlüsseln (AWS CLI)

Führen Sie die folgenden Befehle aus, um die IAM-Benutzerzugriffsschlüssel von aus zu verwalten. AWS CLI

- Zum Erstellen eines Zugriffsschlüssels: `aws iam create-access-key`
- So aktivieren oder deaktivieren Sie einen Zugriffsschlüssel: `aws iam update-access-key`
- Zum Auflisten der Zugriffsschlüssel eines Benutzers: `aws iam list-access-keys`
- Zur Feststellung, wann ein Zugriffsschlüssel zuletzt genutzt wurde: `aws iam get-access-key-last-used`
- Zum Löschen eines Zugriffsschlüssels: `aws iam delete-access-key`

Verwaltung von Zugriffsschlüsseln (AWS API)

Rufen Sie die folgenden Operationen auf, um die Zugriffsschlüssel eines IAM-Benutzers über die AWS API zu verwalten.

- Zum Erstellen eines Zugriffsschlüssels: `CreateAccessKey`

- So aktivieren oder deaktivieren Sie einen Zugriffsschlüssel: `UpdateAccessKey`
- Zum Auflisten der Zugriffsschlüssel eines Benutzers: `ListAccessKeys`
- Zur Feststellung, wann ein Zugriffsschlüssel zuletzt genutzt wurde: `GetAccessKeyLastUsed`
- Zum Löschen eines Zugriffsschlüssels: `DeleteAccessKey`

Aktualisierung der Zugriffsschlüssel

Als bewährte Sicherheitsmethode empfehlen wir Ihnen, die IAM-Benutzerzugriffsschlüssel bei Bedarf zu aktualisieren, beispielsweise wenn ein Mitarbeiter Ihr Unternehmen verlässt. IAM-Benutzer können ihre eigenen Zugriffsschlüssel aktualisieren, wenn ihnen die erforderlichen Berechtigungen gewährt wurden.

Weitere Informationen zum Gewähren von Berechtigungen für IAM-Benutzer zum Aktualisieren ihrer eigenen Zugriffsschlüssel finden Sie unter AWS: Ermöglicht IAM-Benutzern, ihr eigenes Passwort, ihre eigenen Zugangsschlüssel und öffentlichen SSH-Schlüssel auf der Seite Sicherheitsanmeldedaten zu verwalten. Sie können auch eine Passwortrichtlinie auf Ihr Konto anwenden, um zu verlangen, dass alle Ihre IAM-Benutzer ihre Passwörter regelmäßig aktualisieren und festlegen, wie oft sie dies tun müssen. Weitere Informationen finden Sie unter Einrichten einer Kontopasswortrichtlinie für IAM-Benutzer.

Themen

- Aktualisieren von Zugriffsschlüsseln für IAM-Benutzer (Konsole)
- Aktualisieren der Zugriffsschlüssel (AWS CLI)
- Aktualisierung der Zugriffsschlüssel (AWS API)

Aktualisieren von Zugriffsschlüsseln für IAM-Benutzer (Konsole)

Sie können Zugriffsschlüssel über die AWS Management Console aktualisieren.

So aktualisieren Sie Zugriffsschlüssel für einen IAM-Benutzer, ohne Ihre Anwendungen zu unterbrechen (Konsole)

1. Erstellen Sie einen zweiten, solange der erste Zugriffsschlüssel aktiv ist.
    a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
    b. Klicken Sie im Navigationsbereich auf Users (Benutzer).

c. Wählen Sie den Namen des gewünschten Benutzers und dann die Registerkarte Security credentials (Sicherheitsanmeldeinformationen).

d. Wählen Sie im Abschnitt Access keys (Zugriffsschlüssel) Create access key (Zugriffsschlüssel erstellen). Wählen Sie auf der Seite Access key best practices & alternatives (Bewährte Methoden und Alternativen zu Zugriffsschlüsseln) die Option Other (Andere) und anschließend Next (Weiter) aus.

e. (Optional) Legen Sie einen Beschreibungs-Tag-Wert für den Zugriffsschlüssel fest, um diesem IAM-Benutzer ein Tag-Schlüssel-Wert-Paar hinzuzufügen. Auf diese Weise können Sie Zugriffsschlüssel leichter identifizieren und aktualisieren. Der Tag-Schlüssel ist auf die Zugriffsschlüssel-ID festgelegt. Der Tag-Wert ist auf die von Ihnen angegebene Beschreibung des Zugriffsschlüssels festgelegt. Wenn Sie fertig sind, wählen Sie Create access key (Zugriffsschlüssel erstellen) aus.

f. Wählen Sie auf der Seite Retrieve access keys (Zugriffsschlüssel abrufen) entweder Show (Anzeigen), um den Wert des geheimen Zugriffsschlüssels Ihres Benutzers anzuzeigen, oder Download .csv file (CSV-Datei herunterladen). Das ist Ihre einzige Gelegenheit, Ihren geheimen Zugriffsschlüssel zu speichern. Nachdem Sie Ihren geheimen Zugriffsschlüssel an einem sicheren Ort gespeichert haben, wählen Sie Done (Fertig) aus.

   Wenn Sie einen Zugriffsschlüssel erstellen, ist das Schlüsselpaar standardmäßig aktiv, und Ihr Benutzer kann es sofort verwenden. Nun verfügt der Benutzer über zwei aktive Zugriffsschlüssel.

2. Aktualisieren Sie alle Anwendungen und Tools, damit der neue Zugriffsschlüssel verwendet wird.
3. Stellen Sie fest, ob der erste Zugriffsschlüssel noch verwendet wird, indem Sie die Informationen Last used (Zuletzt verwendet) für den ältesten Zugriffsschlüssel prüfen. Es empfiehlt sich, einige Tage zu warten und dann den ältesten verwendeten Zugriffsschüssel zu prüfen, bevor Sie weitere Schritte unternehmen.
4. Auch wenn der Wert in den Informationen Last used (Zuletzt verwendet) angibt, dass der alte Schlüssel zu keiner Zeit verwendet worden ist, empfehlen wir, den ersten Zugriffsschlüssel nicht sofort zu löschen. Wählen Sie stattdessen Actions (Aktionen) und dann Deactivate (Deaktivieren) aus, um den ersten Zugriffsschlüssel zu deaktivieren.
5. Verwenden Sie nur den neuen Zugriffsschlüssel, um zu bestätigen, dass Ihre Anwendungen funktionieren. Alle Anwendungen und Tools, die immer noch den ursprünglichen Zugriffsschlüssel verwenden, funktionieren ab diesem Zeitpunkt nicht mehr, da sie keinen Zugriff mehr auf AWS Ressourcen haben. Wenn Sie so eine Anwendung oder so ein Tool finden,

können Sie den ersten Zugriffsschlüssel reaktivieren. Kehren Sie dann zu Step 3 zurück und aktualisieren Sie diese Anwendung, damit sie den neuen Schlüssel verwendet.

6. Nachdem Sie über einen gewissen Zeitraum sichergestellt haben, dass alle Anwendungen und Tools aktualisiert wurden, können Sie den ersten Zugriffsschlüssel löschen:

   a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   b. Klicken Sie im Navigationsbereich auf Users (Benutzer).

   c. Wählen Sie den Namen des gewünschten Benutzers und dann die Registerkarte Security credentials (Sicherheitsanmeldeinformationen).

   d. Suchen Sie im Abschnitt Access keys (Zugriffsschlüssel) nach dem Zugriffsschlüssel, den Sie löschen möchten, und wählen Sie dann Actions (Aktionen) und anschließend Delete (Löschen). Folgen Sie den Anweisungen im Dialogfeld, um zuerst zu Deactivate (Deaktivieren) und dann den Löschvorgang zu bestätigen.

So stellen Sie fest, welche Zugriffsschlüssel aktualisiert oder gelöscht werden müssen (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 auf Users (Benutzer).
3. Falls erforderlich, fügen Sie die Spalte Access key age (Zugriffsschlüsselalter) zur Tabelle "Benutzer" hinzu, indem Sie die folgenden Schritte ausführen:

   a. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ( ).

   b. Wählen Sie unter Manage columns (Spalten verwalten) die Option Access key age (Zugriffsschlüsselalter).

   c. Klicken Sie auf Close (Schließen), um zur Liste der Benutzer zurückzukehren.

4. Die Spalte Access key age (Zugriffsschlüsselalter) zeigt die Anzahl der Tage an, seit der älteste aktive Zugriffsschlüssel erstellt wurde. Mithilfe dieser Informationen können Sie Benutzer mit Zugriffsschlüsseln finden, die möglicherweise aktualisiert oder gelöscht werden müssen. Für Benutzer ohne Zugriffsschlüssel wird in der Spalte None (Keiner) angezeigt.

Aktualisieren der Zugriffsschlüssel (AWS CLI)

Sie können Zugriffsschlüssel über den AWS Command Line Interface aktualisieren.

So aktualisieren Sie Zugriffsschlüssel, ohne Ihre Anwendungen zu unterbrechen (AWS CLI)

1. Erstellen Sie einen zweiten, standardmäßig aktiven Zugriffsschlüssel, solange der erste Zugriffsschlüssel aktiv ist. Führen Sie den folgenden Befehl aus:

   - `aws iam create-access-key`

   Nun verfügt der Benutzer über zwei aktive Zugriffsschlüssel.
2. Aktualisieren Sie alle Anwendungen und Tools, damit der neue Zugriffsschlüssel verwendet wird.
3. Stellen Sie mit folgendem Befehl fest, ob der erste Zugriffsschlüssel noch verwendet wird:

   - `aws iam get-access-key-last-used`

   Es empfiehlt sich, einige Tage zu warten und dann den ältesten verwendeten Zugriffsschüssel zu prüfen, bevor Sie weitere Schritte unternehmen.
4. Auch wenn der Schritt Step 3 angibt, dass der alte Schlüssel nicht verwendet wird, empfehlen wir, den ersten Zugriffsschlüssel nicht zu sofort löschen. Ändern Sie stattdessen den Status des ersten Zugriffsschlüssels mit dem folgenden Befehl auf `Inactive`:

   - `aws iam update-access-key`
5. Verwenden Sie nur den neuen Zugriffsschlüssel, um zu bestätigen, dass Ihre Anwendungen funktionieren. Alle Anwendungen und Tools, die immer noch den ursprünglichen Zugriffsschlüssel verwenden, funktionieren ab diesem Zeitpunkt nicht mehr, da sie keinen Zugriff mehr auf AWS Ressourcen haben. Wenn Sie so eine Anwendung oder so ein Tool feststellen, können Sie den Status zurück zu `Active` wechseln, um den ersten Zugriffsschlüssel zu reaktivieren. Kehren Sie dann zum Schritt Step 2 zurück und aktualisieren Sie diese Anwendung, damit sie den neuen Schlüssel verwendet.
6. Nachdem Sie über einen gewissen Zeitraum sichergestellt haben, dass alle Anwendungen und Tools aktualisiert wurden, können Sie den ersten Zugriffsschlüssel mit diesem Befehl löschen:

   - `aws iam delete-access-key`

Aktualisierung der Zugriffsschlüssel (AWS API)

Sie können die Zugriffsschlüssel mithilfe der AWS API aktualisieren.

Um Zugriffsschlüssel zu aktualisieren, ohne Ihre Anwendungen zu unterbrechen (AWS API)

1. Erstellen Sie einen zweiten, standardmäßig aktiven Zugriffsschlüssel, solange der erste Zugriffsschlüssel aktiv ist. Rufen Sie die folgende Operation auf:

   - `CreateAccessKey`

   Nun verfügt der Benutzer über zwei aktive Zugriffsschlüssel.
2. Aktualisieren Sie alle Anwendungen und Tools, damit der neue Zugriffsschlüssel verwendet wird.
3. Stellen Sie durch Aufrufen der folgendem Operation fest, ob der erste Zugriffsschlüssel noch verwendet wird:

   - `GetAccessKeyLastUsed`

   Es empfiehlt sich, einige Tage zu warten und dann den ältesten verwendeten Zugriffsschüssel zu prüfen, bevor Sie weitere Schritte unternehmen.
4. Auch wenn der Schritt Step 3 angibt, dass der alte Schlüssel nicht verwendet wird, empfehlen wir, den ersten Zugriffsschlüssel nicht zu sofort löschen. Ändern Sie stattdessen den Status des ersten Zugriffsschlüssels in `Inactive`, indem Sie die Operation aufrufen:

   - `UpdateAccessKey`
5. Verwenden Sie nur den neuen Zugriffsschlüssel, um zu bestätigen, dass Ihre Anwendungen funktionieren. Alle Anwendungen und Tools, die immer noch den ursprünglichen Zugriffsschlüssel verwenden, funktionieren zu diesem Zeitpunkt nicht mehr, da sie keinen Zugriff mehr auf AWS Ressourcen haben. Wenn Sie so eine Anwendung oder so ein Tool feststellen, können Sie den Status zurück zu `Active` wechseln, um den ersten Zugriffsschlüssel zu reaktivieren. Kehren Sie dann zum Schritt Step 2 zurück und aktualisieren Sie diese Anwendung, damit sie den neuen Schlüssel verwendet.
6. Nachdem Sie über einen gewissen Zeitraum sichergestellt haben, dass alle Anwendungen und Tools aktualisiert wurden, können Sie den ersten Zugriffsschlüssel durch Aufrufen dieser Operation löschen:

   - `DeleteAccessKey`

Zugriffsschlüssel sichern

Jeder, der über Ihre Zugriffsschlüssel verfügt, hat denselben Zugriff auf Ihre AWS Ressourcen wie Sie. Daher AWS unternimmt er erhebliche Anstrengungen, um Ihre Zugangsschlüssel zu schützen, und gemäß unserem Modell der geteilten Verantwortung sollten Sie das auch tun.

Erweitern Sie die folgenden Abschnitte, um Anleitungen zum Schutz Ihrer Zugriffsschlüssel zu erhalten.

> **Note**
>
> Ihre Organisation hat unter Umständen andere Sicherheitsanforderungen und -richtlinien als die in diesem Thema beschriebenen. Die hier gemachten Vorschläge dienen lediglich als allgemeine Orientierungshilfe.

Entfernen Sie die Zugriffsschlüssel (oder generieren Sie sie nicht) Root-Benutzer des AWS-Kontos

Eine der besten Methoden, Ihr Konto zu schützen, besteht darin, für Ihr Root-Benutzer des AWS-Kontos gar keinen Zugriffsschlüssel zu verwenden. Sofern Sie nicht über Root-Benutzer-Zugriffsschlüssel verfügen müssen (was selten vorkommt), ist es am besten, diese nicht zu generieren. Erstellen Sie stattdessen einen Administratorbenutzer AWS IAM Identity Center für die täglichen Verwaltungsaufgaben. Informationen zum Erstellen eines Administratorbenutzers in IAM Identity Center finden Sie unter Erste Schritte im IAM Identity Center-Benutzerhandbuch.

Wenn Sie bereits über Root-Benutzer-Zugriffsschlüssel für Ihr Konto verfügen, empfehlen wir Folgendes: Suchen Sie in Ihren Anwendungen nach Stellen, an denen Sie derzeit Zugriffsschlüssel verwenden (falls vorhanden), und ersetzen Sie die Root-Benutzer-Zugriffsschlüssel durch IAM-Benutzerzugriffsschlüssel. Deaktivieren und entfernen Sie dann die Root-Benutzer-Zugriffsschlüssel. Weitere Informationen zum Aktualisieren der Zugriffsschlüsseln finden Sie unter Aktualisierung der Zugriffsschlüssel

Verwenden Sie temporäre Sicherheitsanmeldeinformationen (IAM-Rollen) anstelle von Langzeit-Zugriffsschlüsseln

In vielen Fällen benötigen Sie keine langfristigen Zugriffsschlüssel, die nie ablaufen (wie es bei IAM-Benutzern der Fall ist). Erstellen Sie stattdessen IAM-Rollen und generieren Sie temporäre Sicherheitsanmeldeinformationen. Temporäre Sicherheitsanmeldeinformationen bestehen aus einer

Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel, enthalten aber auch ein Sicherheits-Token, das angibt, wann die Anmeldeinformationen ablaufen.

Langzeit-Zugriffsschlüssel, wie etwa solche für IAM-Benutzer und -Root-Benutzer, bleiben so lange gültig, bis Sie sie manuell widerrufen. Temporäre Sicherheitsanmeldedaten, die Sie über IAM-Rollen und andere Funktionen von erhalten haben, AWS Security Token Service laufen jedoch nach kurzer Zeit ab. Verwenden Sie temporäre Sicherheitsanmeldeinformationen, um das Risiko für den Fall zu verringern, dass Anmeldeinformationen versehentlich kompromittiert werden.

Verwenden Sie eine IAM-Rolle und temporäre Sicherheitsanmeldeinformationen in den folgenden Szenarien:

- Sie haben eine Anwendung oder AWS CLI Skripte, die auf einer Amazon EC2 EC2-Instance ausgeführt werden. Verwenden Sie Zugriffsschlüssel nicht direkt in Ihrer Anwendung. Übergeben Sie keine Zugriffsschlüssel an die Anwendung, betten Sie sie nicht in die Anwendung ein und lassen Sie die Anwendung keine Zugriffsschlüssel beliebiger Quellen lesen. Definieren Sie stattdessen eine IAM-Rolle mit entsprechenden Berechtigungen für Ihre Anwendung und starten Sie die Amazon Elastic Compute Cloud (Amazon EC2)-Instance mit Rollen für EC2. Dadurch wird der Amazon-EC2-Instance eine IAM-Rolle zugeordnet. Durch diese Vorgehensweise erhält die Anwendung auch temporäre Sicherheitsanmeldeinformationen, die sie wiederum für programmgesteuerte Aufrufe an AWS verwenden kann. Die AWS SDKs und die AWS Command Line Interface (AWS CLI) können automatisch temporäre Anmeldeinformationen von der Rolle abrufen.
- Sie müssen kontenübergreifenden Zugriff gewähren. Verwenden Sie eine IAM-Rolle zum Einrichten von Vertrauensstellungen zwischen Konten und erteilen Sie Benutzern in einem Konto eingeschränkte Zugriffsberechtigungen für das vertrauenswürdige Konto. Weitere Informationen finden Sie unter Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen.
- Sie verfügen über eine mobile App. Betten Sie keine Zugriffsschlüssel in die App ein, auch nicht in verschlüsselten Speichern. Verwenden Sie stattdessen Amazon Cognito zum Verwalten der Benutzeridentitäten in Ihrer App. Dieser Service ermöglicht Ihnen die Authentifizierung von Benutzern, die Login with Amazon, Facebook, Google oder einen beliebigen OpenID Connect (OIDC)-kompatiblen Identitätsanbieter nutzen. Anschließend können Sie den Anmeldeinformationsanbieter von Amazon Cognito verwenden, um Anmeldeinformationen zu verwalten, die Ihre App für Anforderungen an AWS nutzt.
- Sie möchten sich zu SAML 2.0 AWS zusammenschließen und Ihr Unternehmen unterstützt es. Falls Ihre Organisation über einen Identitätsanbieter verfügt, der SAML 2.0 unterstützt, konfigurieren Sie den Anbieter für SAML. Sie können SAML verwenden, um

Authentifizierungsinformationen mit temporären Sicherheitsanmeldedaten auszutauschen AWS und diese zurückzugewinnen. Weitere Informationen finden Sie unter SAML 2.0-Verbund.

- Sie möchten sich mit einem lokalen Identitätsspeicher verbinden AWS und Ihre Organisation verfügt über einen lokalen Identitätsspeicher. Wenn sich Benutzer innerhalb Ihrer Organisation authentifizieren können, können Sie eine Anwendung schreiben, die ihnen temporäre Sicherheitsanmeldedaten für den Zugriff auf Ressourcen ausstellt. AWS Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

> **Note**
>
> Verwenden Sie eine Amazon EC2 EC2-Instance mit einer Anwendung, die programmgesteuerten Zugriff AWS auf Ressourcen benötigt? Falls ja, verwenden Sie IAM-Rollen für EC2.

IAM-Benutzerzugriffsschlüssel richtig verwalten

Wenn Sie Zugriffsschlüssel für den programmatischen Zugriff erstellen müssen AWS, erstellen Sie diese für IAM-Benutzer und gewähren Sie den Benutzern nur die Berechtigungen, die sie benötigen.

Beachten Sie diese Vorsichtsmaßnahmen, um die Zugriffsschlüssel von IAM-Benutzern zu schützen:

- Betten Sie Zugriffsschlüssel nicht direkt in den Code ein. Die AWS -SDKs und AWS -Befehlszeilen-Tools ermöglichen es Ihnen, Zugriffsschlüssel an bekannten Speicherorten abzulegen, sodass Sie sie nicht als Code behalten müssen.

  Legen Sie Zugriffsschlüssel an einem der folgenden Orte ab:

  - Die Datei mit den AWS Anmeldeinformationen. Die AWS SDKs verwenden AWS CLI automatisch die Anmeldeinformationen, die Sie in der AWS Anmeldeinformationsdatei speichern.

    Informationen zur Verwendung der AWS Anmeldeinformationsdatei finden Sie in der Dokumentation zu Ihrem SDK. Beispiele hierfür sind Set AWS Credentials and Region im AWS SDK for Java Developer Guide und Configuration and Credentials Files im AWS Command Line Interface User Guide.

    Um Anmeldeinformationen für AWS SDK for .NET und zu speichern AWS Tools for Windows PowerShell, empfehlen wir, den SDK Store zu verwenden. Weitere Informationen finden Sie unter Verwenden des SDK-Speichers im AWS SDK for .NET -Entwicklerhandbuch.

  - Umgebungsvariablen. Wählen Sie in einem System mit mehreren Mandanten Benutzerumgebungsvariablen und keine Systemumgebungsvariablen aus.

    Weitere Informationen zur Verwendung von Umgebungsvariablen zum Speichern von Anmeldeinformationen finden Sie unter Umgebungsvariablen im AWS Command Line Interface - Benutzerhandbuch.

- Verwenden Sie unterschiedliche Zugriffsschlüssel für unterschiedliche Anwendungen. Wenn Sie dies tun, können Sie die Berechtigungen isolieren und die Zugriffsschlüssel für einzelne Anwendungen widerrufen, wenn ein Zugriffsschlüssel kompromittiert wurde. Mit separaten Zugriffsschlüsseln für verschiedene Anwendungen werden auch eindeutige Einträge in AWS CloudTrail-Protokolldateien generiert. Mit dieser Konfiguration können Sie leichter feststellen, welche Anwendung bestimmte Aktionen ausgeführt hat.
- Aktualisieren Sie die Zugriffsschlüssel bei Bedarf. Wenn das Risiko besteht, dass der Zugriffsschlüssel kompromittiert werden könnte, aktualisieren Sie den Zugriffsschlüssel und löschen Sie den vorherigen Zugriffsschlüssel. Details hierzu finden Sie unter Aktualisierung der Zugriffsschlüssel
- Entfernen Sie nicht verwendete Zugriffsschlüssel. Wenn ein Benutzer Ihre Organisation verlässt, entfernen Sie den entsprechenden IAM-Benutzer, damit dieser nicht mehr auf Ihre Ressourcen zugreifen kann. Um herauszufinden, wann ein Zugriffsschlüssel zuletzt verwendet wurde, verwenden Sie die `GetAccessKeyLastUsed`API (AWS CLI command: `aws iam get-access-key-last-used`).
- Verwenden Sie temporäre Anmeldeinformationen und konfigurieren Sie die Multi-Faktor-Authentifizierung für Ihre sensibelsten API-Vorgänge. Mit IAM-Richtlinien können Sie angeben, welche API-Operationen ein Benutzer aufrufen darf. In einigen Fällen möchten Sie möglicherweise die zusätzliche Sicherheit nutzen, die darin besteht, dass Benutzer mit AWS MFA authentifiziert werden müssen, bevor Sie ihnen erlauben, besonders vertrauliche Aktionen auszuführen. Sie verfügen beispielsweise über eine Richtlinie, die dem Benutzer gestattet, die Amazon EC2 Aktionen `RunInstances`, `DescribeInstances` und `StopInstances` auszuführen. Möglicherweise möchten Sie jedoch eine destruktive Aktion einschränken `TerminateInstances` und sicherstellen, dass Benutzer diese Aktion nur ausführen können, wenn sie sich mit einem AWS MFA-Gerät authentifizieren. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs.

Greifen Sie mit Zugangsschlüsseln auf die mobile App zu AWS

Mit der AWS mobilen App können Sie auf eine begrenzte Anzahl von AWS Diensten und Funktionen zugreifen. Mit der mobilen App können Sie auch unterwegs auf Vorfälle reagieren. Unter AWS Console Mobile Application können Sie die App herunterladen und weitere Informationen erhalten.

Sie können sich mit Ihrem Konsolenpasswort oder Ihren Zugriffsschlüsseln bei der mobilen App anmelden. Verwenden Sie als bewährte Methode keine Stammbenutzer-Zugriffsschlüssel. Stattdessen empfehlen wir dringend, zusätzlich zur Verwendung eines Kennworts oder einer biometrischen Sperre auf Ihrem Mobilgerät einen IAM-Benutzer speziell für die Verwaltung von AWS Ressourcen mithilfe der mobilen App einzurichten. Wenn Sie Ihr Mobilgerät verlieren, können Sie den Zugriff des IAM-Benutzers entfernen.

So melden Sie sich mit Zugriffsschlüsseln an (mobile App)

1. Öffnen Sie die App auf Ihrem mobilen Gerät.
2. Wenn Sie dem Gerät zum ersten Mal eine Identität hinzufügen, wählen Sie Identität hinzufügen und dann Zugriffsschlüssel aus.

   Wenn Sie sich bereits mit einer anderen Identität angemeldet haben, wählen Sie das Menüsymbol und dann Identität wechseln aus. Wählen Sie dann Als andere Identität anmelden und dann Zugriffsschlüssel aus.
3. Geben Sie auf der Seite Zugriffsschlüssel Ihre Informationen ein:

   - Zugriffsschlüssel-ID: Geben Sie Ihre Zugriffsschlüssel-ID ein.
   - Geheimer Zugriffsschlüssel: Geben Sie Ihren geheimen Zugriffsschlüssel ein.
   - Identitätsname: Geben Sie den Namen der Identität ein, der in der mobilen App angezeigt wird. Dieser muss nicht mit Ihrem IAM-Benutzernamen übereinstimmen.
   - Identitäts-PIN: Erstellen Sie eine persönliche Identifizierungsnummer (PIN), die Sie bei zukünftigen Anmeldungen verwenden.

   > **Note**
   >
   > Wenn Sie die Biometrie für die AWS mobile App aktivieren, werden Sie aufgefordert, anstelle der PIN Ihren Fingerabdruck oder Ihre Gesichtserkennung zur Überprüfung zu verwenden. Wenn die Biometrie fehlschlägt, werden Sie möglicherweise stattdessen zur Eingabe der PIN aufgefordert.

4. Wählen Sie Überprüfen und Hinzufügen von Schlüsseln.

   Sie können nun über die mobile App auf eine ausgewählte Gruppe Ihrer Ressourcen zugreifen.

Ähnliche Informationen

Die folgenden Themen enthalten Anleitungen zur Einrichtung der AWS SDKs und AWS CLI zur Verwendung von Zugriffsschlüsseln:

- Legen Sie die AWS Anmeldeinformationen und die Region im AWS SDK for Java Entwicklerhandbuch fest
- Verwendung des SDK-Speichers im AWS SDK for .NET -Entwicklerhandbuch
- Bereitstellung von Anmeldeinformationen für das SDK im AWS SDK for PHP -Entwicklerhandbuch
- Konfiguration in der Boto 3-Dokumentation (AWS SDK für Python)
- Verwendung von AWS -Anmeldeinformationen im AWS Tools for Windows PowerShell -Benutzerhandbuch
- Konfigurations- und Anmeldeinformationsdateien im AWS Command Line Interface -Benutzerhandbuch
- Gewährung des Zugriffs mithilfe einer IAM-Rolle im AWS SDK for .NET -Entwicklerhandbuch
- Konfigurieren von IAM-Rollen für Amazon EC2 in der AWS SDK for Java 2.x

Überwachen von Zugriffsschlüsseln

Sie können die AWS Zugriffsschlüssel in Ihrem Code überprüfen, um festzustellen, ob die Schlüssel von einem Konto stammen, das Sie besitzen. Sie können eine Zugriffsschlüssel-ID mithilfe des `aws sts get-access-key-info` AWS CLI Befehls oder der `GetAccessKeyInfo` AWS API-Operation übergeben.

Die AWS API-Operationen AWS CLI und geben die ID der Person zurück, AWS-Konto zu der der Zugriffsschlüssel gehört. Zugriffsschlüssel-IDs, die mit `AKIA` beginnen, sind langfristige Anmeldeinformationen für einen IAM-Benutzer oder Root-Benutzer des AWS-Kontos. Bei Zugriffsschlüssel-IDs, die mit beginnen, `ASIA` handelt es sich um temporäre Anmeldeinformationen, die mithilfe von AWS STS Vorgängen erstellt werden. Wenn das Konto in der Antwort Ihnen gehört, können Sie sich als Stammbenutzer anmelden und Ihre Stammbenutzer-Zugriffsschlüssel überprüfen. Anschließend können Sie einen Anmeldeinformationsbericht abrufen, um zu erfahren, welcher IAM-Benutzer die Schlüssel besitzt. Um zu erfahren, wer die temporären

Anmeldeinformationen für einen `ASIA` Zugriffsschlüssel angefordert hat, sehen Sie sich die AWS STS Ereignisse in Ihren CloudTrail Protokollen an.

Aus Sicherheitsgründen können Sie in den AWS CloudTrail Protokollen nachlesen, wer eine Aktion in ausgeführt hat AWS. Sie können den `sts:SourceIdentity`-Bedingungsschlüssel in der Rollenvertrauensrichtlinie verwenden, damit Benutzer einen Sitzungsnamen angeben müssen, wenn sie eine Rolle übernehmen. Sie können beispielsweise verlangen, dass IAM-Benutzer ihren eigenen Benutzernamen als Sitzungsnamen angeben. Auf diese Weise können Sie feststellen, welcher Benutzer eine bestimmte Aktion in AWS ausgeführt hat. Weitere Informationen finden Sie unter `sts:SourceIdentity`.

Diese Operation gibt nicht den Status des Zugriffsschlüssels an. Der Schlüssel kann aktiv, inaktiv oder gelöscht sein. Aktive Schlüssel verfügen möglicherweise nicht über Berechtigungen zum Ausführen einer Operation. Die Bereitstellung eines gelöschten Zugriffsschlüssels gibt möglicherweise einen Fehler zurück, der besagt, dass der Schlüssel nicht vorhanden ist.

Zurücksetzen verlorener oder vergessener Passwörter oder Zugangsschlüssel für AWS

> **Important**
>
> Haben Sie Probleme bei der Anmeldung? AWS Stellen Sie sicher, dass Sie sich auf der richtigen AWS -Anmeldeseite für Ihren Benutzertyp befinden. Wenn Sie der Root-Benutzer des AWS-Kontos (Kontoinhaber) sind, können Sie sich AWS mit den Anmeldeinformationen anmelden, die Sie bei der Erstellung des eingerichtet haben AWS-Konto. Wenn Sie ein IAM-Benutzer sind, kann Ihr Kontoadministrator Ihnen die Anmeldeinformationen bereitstellen, mit denen Sie sich bei AWS anmelden können. Wenn Sie Support anfordern müssen, verwenden Sie nicht den Feedback-Link auf dieser Seite, da das Formular beim AWS Dokumentationsteam eingegangen ist, nicht AWS Support. Wählen Sie stattdessen auf der Seite Kontakt die Option Sie können sich noch immer nicht in Ihrem AWS -Konto anmelden und wählen Sie dann eine der verfügbaren Support-Optionen.

Auf der Hauptanmeldeseite müssen Sie Ihre E-Mail-Adresse eingeben, um sich als Stammbenutzer anzumelden, oder Ihre Konto-ID eingeben, um sich als IAM-Benutzer anzumelden. Sie können Ihr Passwort nur auf der Anmeldeseite angeben, die Ihrem Benutzertyp entspricht. Weitere Informationen dazu finden Sie unter Anmelden bei der AWS Management Console.

Wenn Sie sich auf der richtigen Anmeldeseite befinden und Ihre Passwörter oder Zugriffsschlüssel verlieren oder vergessen, können Sie diese nicht aus IAM abrufen. Sie können sie stattdessen mithilfe der folgenden Methoden zurücksetzen:

- Root-Benutzer des AWS-Kontos Passwort — Wenn Sie Ihr Root-Benutzerpasswort vergessen haben, können Sie das Passwort über den zurücksetzen AWS Management Console. Weitere Informationen finden Sie unter the section called “Zurücksetzen eines verlorenen oder vergessenen Stammbenutzer-Passworts” an späterer Stelle in diesem Thema.
- AWS-Konto Zugangsschlüssel — Wenn Sie Ihre Kontozugriffsschlüssel vergessen haben, können Sie neue Zugangsschlüssel erstellen, ohne die vorhandenen Zugangsschlüssel zu deaktivieren. Wenn Sie die vorhandenen Schlüssel nicht verwenden, können Sie diese löschen. Details dazu finden Sie unter Erstellen von Zugriffsschlüsseln für den Stammbenutzer und Löschen von Zugriffsschlüsseln für den Stammbenutzer.
- IAM-BenutzerPasswort – Wenn Sie ein IAM-Benutzer sind und Ihr Passwort vergessen haben, wenden Sie sich an Ihren Administrator, um Ihr Passwort zurückzusetzen. Mehr zur Verwaltung Ihres Passwortes durch einen Administrator finden Sie unter Verwalten von Passwörtern für IAM-Benutzer.
- IAM-Benutzerzugriffsschlüssel – Wenn Sie ein IAM-Benutzer sind und Sie Ihre Zugriffsschlüssel vergessen haben, benötigen Sie neue Zugriffsschlüssel. Wenn Sie über die Berechtigung zum Erstellen von Zugriffsschlüsseln verfügen, finden Sie eine Anleitung zum Erstellen unter Verwalten von Zugriffsschlüsseln (Konsole). Wenn Sie nicht über die erforderlichen Berechtigungen verfügen, wenden Sie sich an Ihren Administrator, um neue Zugriffsschlüssel zu erstellen. Wenn Sie noch mit Ihren alten Schlüsseln arbeiten, bitten Sie Ihren Administrator darum, die alten Schlüssel zu löschen. Mehr zur Verwaltung Ihrer Zugriffsschlüssel durch einen Administrator finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer.

Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS

Follow us on Twitter

Um die Sicherheit zu erhöhen, empfehlen wir Ihnen, die Multi-Faktor-Authentifizierung (MFA) zu konfigurieren, um Ihre AWS Ressourcen zu schützen. Sie können MFA für die Root-Benutzer des AWS-Kontos und IAM-Benutzer aktivieren. Wenn Sie MFA für den Stammbenutzer aktivieren, wirkt sich das nur auf die Anmeldeinformationen des Stammbenutzers. IAM-Benutzer im Konto sind unabhängige Identitäten mit eigenen Anmeldeinformationen und einer jeweils individuellen MFA-Konfiguration.

Sie können bis zu acht MFA-Geräte einer beliebigen Kombination der derzeit unterstützten MFA-Typen für Ihren Root-Benutzer des AWS-Kontos und Ihre IAM-Benutzer registrieren. Weitere Informationen zu unterstützten MFA-Typen finden Sie unter Verfügbare MFA-Typen für IAM-Benutzer. Bei mehreren MFA-Geräten ist nur ein MFA-Gerät erforderlich, um sich mit dem Benutzer anzumelden AWS Management Console oder eine Sitzung über diesen Benutzer zu erstellen. AWS CLI

> **Note**
>
> Wir empfehlen, dass Sie von Ihren menschlichen Benutzern verlangen, dass sie beim Zugriff temporäre Anmeldeinformationen verwenden. AWS Haben Sie darüber nachgedacht, es zu verwenden AWS IAM Identity Center? Sie können IAM Identity Center verwenden, um den Zugriff auf mehrere Konten zentral zu verwalten AWS-Konten und Benutzern MFA-geschützten Single Sign-On-Zugriff auf alle ihnen zugewiesenen Konten von einem Ort aus zu gewähren. Mit IAM Identity Center können Sie Benutzeridentitäten in IAM Identity Center erstellen und verwalten oder einfach eine Verbindung zu Ihrem vorhandenen SAML-2.0-kompatiblen Identitätsanbieter herstellen. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

Verfügbare MFA-Typen für IAM-Benutzer

MFA bietet zusätzliche Sicherheit, da Benutzer zusätzlich zu ihren regulären Anmeldeinformationen eine eindeutige Authentifizierung über einen AWS unterstützten MFA-Mechanismus angeben müssen, wenn sie auf AWS Websites oder Dienste zugreifen. AWS unterstützt die folgenden MFA-Typen: Hauptschlüssel und Sicherheitsschlüssel, virtuelle Authentifikatoranwendungen und Hardware-TOTP-Token.

Hauptschlüssel und Sicherheitsschlüssel

AWS Identity and Access Management unterstützt Hauptschlüssel und Sicherheitsschlüssel für MFA. Basierend auf den FIDO-Standards verwenden Hauptschlüssel Kryptografie mit öffentlichen Schlüsseln, um eine starke, gegen Phishing resistente Authentifizierung zu gewährleisten, die sicherer ist als Passwörter. AWS unterstützt zwei Arten von Hauptschlüsseln: gerätegebundene Hauptschlüssel (Sicherheitsschlüssel) und synchronisierte Hauptschlüssel.

- Sicherheitsschlüssel: Dabei handelt es sich um physische Geräte, wie z. B. a YubiKey, die als zweiter Faktor für die Authentifizierung verwendet werden.

- Synchronisierte Hauptschlüssel: Diese verwenden Anmeldeinformationsmanager von Anbietern wie Google, Apple, Microsoft-Konten und Drittanbieterdiensten wie 1Password, Dashlane und Bitwarden als zweiten Faktor.

Sie können integrierte biometrische Authentifikatoren wie Touch ID auf Apple MacBooks und Windows Hello-Gesichtserkennung auf PCs verwenden, um Ihren Anmeldeinformationsmanager zu entsperren und sich dort anzumelden. AWS Hauptschlüssel werden bei dem von Ihnen ausgewählten Anbieter anhand Ihres Fingerabdrucks, Ihres Gesichts oder Ihrer Geräte-PIN erstellt. Sie können Hauptschlüssel auf Ihren Geräten synchronisieren, um die Anmeldung zu erleichtern und so die Benutzerfreundlichkeit und Wiederherstellbarkeit zu AWS verbessern.

Die FIDO Alliance führt eine Liste aller FIDO-zertifizierten Produkte, die mit den FIDO-Spezifikationen kompatibel sind. Ein einziger Hauptschlüssel oder Sicherheitsschlüssel unterstützt mehrere Root-Benutzerkonten und IAM-Benutzer. Weitere Informationen zur Aktivierung von Hauptschlüsseln und Sicherheitsschlüsseln für einen IAM-Benutzer finden Sie unter. Aktivieren eines Hauptschlüssels oder Sicherheitsschlüssels (Konsole)

Anwendungen für virtuelle Authentifikatoren

Eine virtuelle Authentifizierungsanwendung wird auf einem Telefon oder einem anderen Gerät ausgeführt und emuliert ein physisches Gerät. Virtuelle Authentifizierungs-Apps implementieren den Algorithmus für ein zeitgesteuertes Einmalpasswort (TOTP) und unterstützen mehrere Token auf einem einzigen Gerät. Der Benutzer muss einen gültigen Code vom Gerät eingeben, wenn er bei der Anmeldung dazu aufgefordert wird. Jedes einem Benutzer zugewiesene Token muss eindeutig sein. Ein Benutzer kann zur Authentifizierung keinen Code aus dem Token eines anderen Benutzers eingeben.

Wir empfehlen die Verwendung eines virtuellen MFA-Geräts beim Warten auf die Genehmigung für den Hardware-Kauf oder während Sie warten, bis Ihre Hardware eintrifft. Eine Liste einiger unterstützter Apps, die Sie als virtuelle MFA-Geräte verwenden können, finden Sie unter Multi-Factor Authentication (MFA). Anweisungen zum Einrichten eines virtuellen MFA-Geräts für einen IAM-Benutzer finden Sie unter. Aktivieren eines virtuellen Multi-Factor Authentication (MFA)-Geräts (Konsole)

Hardware-TOTP-Token

Ein Hardwaregerät generiert einen sechsstelligen numerischen Code, der auf dem TOTP-Algorithmus (Time-Based One-Time Password) basiert. Der Benutzer muss während der Anmeldung auf einer zweiten Webseite einen gültigen Code von dem Gerät eingeben. Jedes MFA-Gerät, das einem

Benutzer zugeordnet ist, muss eindeutig sein. Ein Benutzer kann keinen Code von einem MFA-Gerät eines anderen Benutzers eingeben, um sich zu authentifizieren. Informationen zu unterstützten Hardware-MFA-Geräten finden Sie unter Multi-Factor Authentication (MFA). Anweisungen zum Einrichten eines Hardware-TOTP-Tokens für einen IAM-Benutzer finden Sie unter. Aktivieren eines Hardware-TOTP-Tokens (Konsole)

Wenn Sie ein physisches MFA-Gerät verwenden möchten, empfehlen wir die Verwendung von Sicherheitsschlüsseln als Alternative zu Hardware-TOTP-Geräten. Sicherheitsschlüssel bieten den Vorteil, dass keine Batterien erforderlich sind und Phishing-Angriffe verhindert werden. Außerdem unterstützen sie mehrere Root- und IAM-Benutzer auf einem einzigen Gerät, um die Sicherheit zu erhöhen.

> **Note**
>
> SMS-textnachrichtenbasierte MFA – AWS hat die Unterstützung für Multi-Faktor-Authentifizierung (MFA) per SMS eingestellt. Wir empfehlen Kunden, deren IAM-Benutzer MFA auf SMS-Textnachrichtenbasis verwenden, zu einer der folgenden alternativen Methoden zu wechseln: Hauptschlüssel oder Sicherheitsschlüssel, virtuelles (softwarebasiertes) MFA-Gerät oder Hardware-MFA-Gerät. Sie können die Benutzer in Ihrem Konto mit einem zugewiesenen SMS-MFA-Gerät identifizieren. Navigieren Sie zu diesem Zweck zur IAM-Konsole, wählen Sie im Navigationsbereich Users (Benutzer) und suchen Sie nach Benutzern mit SMS in der Tabellenspalte MFA.

Themen

Aktivierung von MFA-Geräten für Benutzer in AWS

Die Schritte zur Konfiguration von MFA hängen vom Typ des verwendeten MFA-Geräts ab.

Themen

Allgemeine Schritte zum Aktivieren von MFA-Geräten

In der folgenden zusammenfassenden Anleitung wird beschrieben, wie Sie MFA konfigurieren und verwenden, und es werden Links zu diesbezüglichen Informationen bereitgestellt.

> **Hinweis**
>
> Weitere Informationen finden Sie auch in diesem englischsprachigen Video How to Setup AWS Multi-Factor Authentication (MFA) and AWS Budget Alerts.

1. Holen Sie sich ein MFA-Gerät, z. B. eines der folgenden. Sie können bis zu acht MFA-Geräte pro Root-Benutzer des AWS-Kontos IAM-Benutzer mit einer beliebigen Kombination der folgenden Typen aktivieren.
   - Ein virtuelles MFA-Gerät, bei dem es sich um eine Softwareanwendung handelt, die mit RFC 6238, einem standardbasierten TOTP-Algorithmus (time based one-time password) kompatibel ist. Sie können die App auf einem Telefon oder einem anderen Gerät installieren. Eine Liste einiger unterstützter Anwendungen, die Sie als virtuelle MFA-Geräte verwenden können, finden Sie unter Multifaktor-Authentifizierung.
   - Ein Hauptschlüssel oder Sicherheitsschlüssel mit einer AWS unterstützten Konfiguration. Die FIDO Alliance führt eine Liste aller FIDO-zertifizierten Produkte, die mit den FIDO-Spezifikationen kompatibel sind.
   - Ein hardwarebasiertes MFA-Gerät von einem Drittanbieter, z. B. ein Token-Gerät. Diese Token werden ausschließlich mit verwendet. AWS-Konten Weitere Informationen finden Sie unter Aktivieren eines Hardware-TOTP-Tokens (Konsole). Sie können nur Token verwenden, deren eindeutige Token-Seeds auf sichere Weise gemeinsam genutzt werden AWS. Token-Seeds sind geheime Schlüssel, die zum Zeitpunkt der Token-Produktion generiert werden. Token, die aus anderen Quellen gekauft wurden, funktionieren mit IAM nicht. Um die Kompatibilität

sicherzustellen, müssen Sie Ihr Hardware-MFA-Gerät über einen der folgenden Links erwerben: OTP-Token oder OTP-Grafikkarte.

2. Aktivieren des MFA-Geräts.
   - Virtuelle oder Hardware-TOTP-Token — Sie können AWS CLI Befehle oder AWS API-Operationen verwenden, um ein virtuelles MFA-Gerät für einen IAM-Benutzer zu aktivieren. Sie können ein MFA-Gerät nicht Root-Benutzer des AWS-Kontos mit der AWS API AWS CLI, Tools für Windows PowerShell oder einem anderen Befehlszeilentool für aktivieren. Sie können jedoch das verwenden, AWS Management Console um ein MFA-Gerät für den Root-Benutzer zu aktivieren.
   - Hauptschlüssel und Sicherheitsschlüssel — Root-Benutzer und IAM-Benutzer mit Hauptschlüsseln oder Sicherheitsschlüsseln können die Aktivierung AWS Management Console nur über die API vornehmen, nicht über die oder. AWS CLI AWS

   Weitere Informationen zum Aktivieren der verschiedenen MFA-Geräte finden Sie auf den folgenden Seiten:
   - Virtuelle MFA-Geräte: Aktivieren eines virtuellen Multi-Factor Authentication (MFA)-Geräts (Konsole)
   - Hauptschlüssel und Sicherheitsschlüssel: Aktivieren eines Hauptschlüssels oder Sicherheitsschlüssels (Konsole)
   - Hardware-TOTP-Token: Aktivieren eines Hardware-TOTP-Tokens (Konsole)
3. Mehrere MFA-Geräte aktivieren (empfohlen)
   - Wir empfehlen, dass Sie mehrere MFA-Geräte für die Root-Benutzer des AWS-Kontos und IAM-Benutzer in Ihrem aktivieren. AWS-Konten Damit können Sie die Sicherheitsstandards in Ihren AWS-Konten erhöhen und die Verwaltung des Zugriffs auf hochprivilegierte Benutzer, wie den Root-Benutzer des AWS-Kontos, vereinfachen.
   - Sie können bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen bei Ihren Root-Benutzer des AWS-Kontos und IAM-Benutzern registrieren. Bei mehreren MFA-Geräten benötigen Sie nur ein MFA-Gerät, um sich AWS CLI als dieser Benutzer anzumelden AWS Management Console oder eine Sitzung über den zu erstellen. Ein IAM-Benutzer muss sich mit einem vorhandenen MFA-Gerät authentifizieren, um ein zusätzliches MFA-Gerät zu aktivieren oder zu deaktivieren.
   - Im Falle eines verlorenen, gestohlenen oder unzugänglichen MFA-Geräts können Sie eines der verbleibenden MFA-Geräte verwenden, um darauf zuzugreifen, AWS-Konto ohne das Wiederherstellungsverfahren durchführen zu müssen. AWS-Konto Wenn ein MFA-Gerät

verloren geht oder gestohlen wird, muss es vom IAM-Prinzipal getrennt werden, mit dem es verknüpft ist.

- Durch die Verwendung mehrerer MFAs können Ihre Mitarbeiter an geografisch verteilten Standorten oder remote arbeiten, hardwarebasiertes MFA für den Zugriff verwenden, AWS ohne den physischen Austausch eines einzelnen Hardwaregeräts zwischen Mitarbeitern koordinieren zu müssen.
- Durch die Verwendung zusätzlicher MFA-Geräte für IAM-Prinzipale können Sie ein oder mehrere MFAs für den täglichen Gebrauch verwenden und gleichzeitig physische MFA-Geräte an einem sicheren physischen Ort wie einem Tresor oder einem Safe für die Sicherung und Redundanz aufbewahren.

4. Verwenden Sie das MFA-Gerät, wenn Sie sich bei AWS -Ressourcen anmelden oder auf sie zugreifen möchten.
   - Hauptschlüssel und Sicherheitsschlüssel — Um auf eine AWS Website zuzugreifen, geben Sie Ihre Anmeldeinformationen ein und tippen Sie dann, je nach Art Ihres Hauptschlüssels, auf den FIDO-Sicherheitsschlüssel, geben Sie eine Geräte-PIN ein oder geben Sie Ihren Fingerabdruck oder Ihr Gesicht ein, wenn Sie dazu aufgefordert werden.
   - Virtuelle MFA-Geräte und Hardware-TOTP-Token — Um auf eine AWS Website zuzugreifen, benötigen Sie zusätzlich zu Ihrem Benutzernamen und Passwort einen MFA-Code vom Gerät.

     Um auf MFA-geschützte API-Operationen zugreifen zu können, benötigen Sie Folgendes:
     - Einen MFA-Code
     - Die Kennung für das MFA-Gerät (die Geräteseriennummer eines physikalischen Geräts oder die ARN eines in AWS-definierten virtuellen Geräts)
     - Die übliche Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel

> **Hinweise**
>
> - Sie können die MFA-Informationen für einen FIDO-Sicherheitsschlüssel nicht an AWS STS API-Operationen weitergeben, um temporäre Anmeldeinformationen anzufordern.
> - Sie können keine AWS CLI Befehle oder AWS API-Operationen verwenden, um FIDO-Sicherheitsschlüssel zu aktivieren.
> - Sie können denselben Namen nicht für mehr als ein Root- oder IAM-MFA-Gerät verwenden.

Weitere Informationen finden Sie unter Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Aktivieren eines Hauptschlüssels oder Sicherheitsschlüssels (Konsole)

Passkeys sind eine Art Multi-Faktor-Authentifizierungsgerät (MFA), mit dem Sie Ihre Ressourcen schützen können. AWS AWS unterstützt synchronisierte Hauptschlüssel und gerätegebundene Hauptschlüssel, auch Sicherheitsschlüssel genannt.

Synchronisierte Hauptschlüssel ermöglichen es IAM-Benutzern, auf ihre FIDO-Anmeldedaten auf vielen ihrer Geräte, auch auf neuen, zuzugreifen, ohne jedes Gerät für jedes Konto neu registrieren zu müssen. Zu den synchronisierten Passkeys gehören Anmeldeinformationsmanager von Erstanbietern wie Google, Apple und Microsoft sowie Anmeldeinformationsmanager von Drittanbietern wie 1Password, Dashlane und Bitwarden als zweiten Faktor. Sie können auch geräteinterne Biometrie (z. B. TouchID, FaceID, Windows Hello) verwenden, um den von Ihnen ausgewählten Anmeldeinformationsmanager für die Verwendung von Hauptschlüsseln zu entsperren.

Alternativ sind gerätegebundene Hauptschlüssel an einen FIDO-Sicherheitsschlüssel gebunden, den Sie an einen USB-Anschluss Ihres Computers anschließen und dann antippen, wenn Sie dazu aufgefordert werden, um den Anmeldevorgang sicher abzuschließen. Wenn Sie einen FIDO-Sicherheitsschlüssel bereits mit anderen Diensten verwenden und dieser über eine AWS unterstützte Konfiguration verfügt (z. B. die Serie YubiKey 5 von Yubico), können Sie ihn auch zusammen verwenden. AWS Andernfalls müssen Sie einen FIDO-Sicherheitsschlüssel erwerben, wenn Sie ihn WebAuthn für MFA in verwenden möchten. AWS Darüber hinaus können FIDO-Sicherheitsschlüssel mehrere IAM- oder Root-Benutzer auf demselben Gerät unterstützen, wodurch ihr Nutzen für die Kontosicherheit verbessert wird. Spezifikationen und Kaufinformationen für beide Gerätetypen finden Sie unter Multifaktor-Authentifizierung.

Sie können bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen bei Ihren Root-Benutzer des AWS-Kontos und IAM-Benutzern registrieren. Bei mehreren MFA-Geräten benötigen Sie nur ein MFA-Gerät, um sich AWS CLI als dieser Benutzer anzumelden AWS Management Console oder eine Sitzung über den zu erstellen. Wir empfehlen, mehrere MFA-Geräte zu registrieren. Sie können beispielsweise einen integrierten Authentifikator registrieren und auch einen Sicherheitsschlüssel registrieren, den Sie an einem physisch sicheren Ort aufbewahren. Wenn Sie Ihren integrierten Authentifikator nicht verwenden können, können Sie Ihren registrierten Sicherheitsschlüssel verwenden. Für Authentifizierungsanwendungen empfehlen wir außerdem, die Cloud-Backup- oder Synchronisierungsfunktion in diesen Apps zu aktivieren. Dadurch vermeiden Sie, dass Sie den Zugriff auf Ihr Konto verlieren, wenn Sie Ihr Gerät mit den Authentifizierungs-Apps verlieren oder beschädigen.

> **Note**
>
> Wir empfehlen, dass Sie menschliche Benutzer auffordern, beim Zugriff auf AWS temporäre Anmeldeinformationen zu verwenden. Ihre Benutzer können sich AWS mit einem Identitätsanbieter verbinden, wo sie sich mit ihren Unternehmensanmeldedaten und MFA-Konfigurationen authentifizieren. Um den Zugriff auf AWS und Geschäftsanwendungen zu verwalten, empfehlen wir die Verwendung von IAM Identity Center. Weitere Informationen finden Sie im IAM Identity Center-Benutzerhandbuch.

Themen

Erforderliche Berechtigungen

Um einen FIDO-Passkey für Ihren eigenen IAM-Benutzer zu verwalten und gleichzeitig sensible MFA-bezogene Aktionen zu schützen, benötigen Sie die Berechtigungen der folgenden Richtlinie:

> **Note**
>
> Die ARN-Werte sind statische Werte und kein Indikator dafür, welches Protokoll zur Registrierung des Authentifikators verwendet wurde. Wir haben U2F als veraltet eingestuft, daher verwenden alle neuen Implementierungen. WebAuthn

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für Ihren eigenen IAM-Benutzer (Konsole)

Sie können einen Hauptschlüssel oder Sicherheitsschlüssel für Ihren eigenen IAM-Benutzer AWS Management Console nur über die API aktivieren, nicht über die oder. AWS CLI AWS Bevor Sie einen Sicherheitsschlüssel aktivieren können, müssen Sie physischen Zugriff auf das Gerät haben.

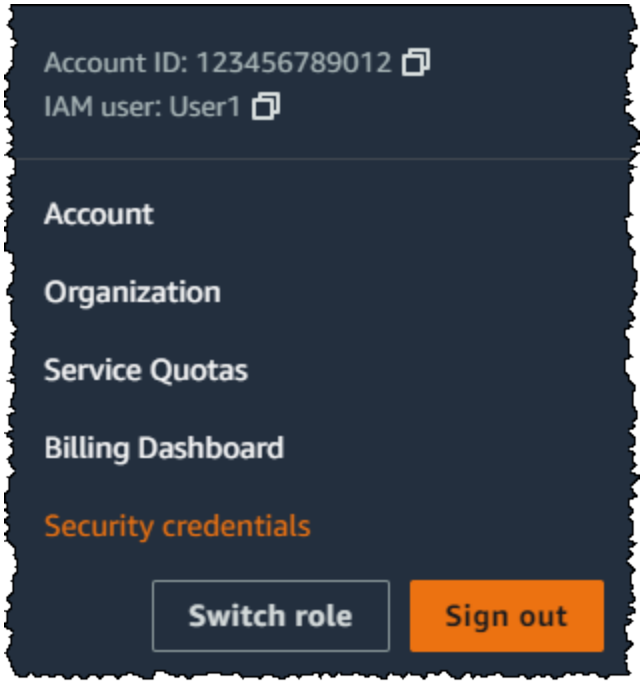
Um einen Hauptschlüssel oder Sicherheitsschlüssel für Ihren eigenen IAM-Benutzer (Konsole) zu aktivieren

1. Verwenden Sie Ihre AWS Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der IAM-Konsole anzumelden.

> **Note**
>
> Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich Ihren IAM-Benutzernamen und Ihre Kontoinformationen zu merken. Wenn Sie sich zuvor als anderer Benutzer angemeldet haben, wählen Sie Melden Sie sich bei einem anderen Konto an Um zur Hauptanmeldeseite zurückzukehren. Von dort aus können Sie Ihre AWS Konto-ID oder Ihren Kontoalias eingeben, um zur IAM-Benutzer-Anmeldeseite für Ihr Konto weitergeleitet zu werden.

Wenden Sie sich an Ihren Administrator, um Ihre AWS-Konto ID zu erhalten.

2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheitsanmeldeinformationen) aus.



3. Wählen Sie auf der Seite des ausgewählten IAM-Benutzers die Registerkarte Sicherheitsanmeldeinformationen aus.
4. Wählen Sie im Abschnitt Multi-Factor Authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) die Option Assign MFA device (MFA-Gerät zuweisen).
5. Geben Sie auf der Seite MFA-Gerätename einen Gerätenamen ein, wählen Sie Hauptschlüssel oder Sicherheitsschlüssel und dann Weiter.

6. Richten Sie auf Gerät einrichten Ihren Hauptschlüssel ein. Erstellen Sie einen Hauptschlüssel mit biometrischen Daten wie Ihrem Gesicht oder Fingerabdruck, mit einer Geräte-PIN oder indem Sie den FIDO-Sicherheitsschlüssel in den USB-Anschluss Ihres Computers stecken und darauf tippen.
7. Folgen Sie den Anweisungen in Ihrem Browser und wählen Sie dann Weiter.

Sie haben jetzt Ihren Hauptschlüssel oder Sicherheitsschlüssel zur Verwendung mit AWS registriert. Hinweise zur Verwendung von MFA mit dem finden Sie AWS Management Console unterVerwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für einen anderen IAM-Benutzer (Konsole)

Sie können einen Hauptschlüssel oder eine Sicherheit für einen anderen IAM-Benutzer AWS Management Console nur über die API aktivieren, nicht über die oder. AWS CLI AWS

Um einen Hauptschlüssel oder eine Sicherheit für einen anderen IAM-Benutzer (Konsole) zu aktivieren

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 auf Users (Benutzer).
3. Wählen Sie unter Benutzer den Namen des Benutzers aus, für den Sie MFA aktivieren möchten.
4. Wählen Sie auf der ausgewählten IAM-Benutzerseite die Registerkarte Security Credentials aus.
5. Wählen Sie im Abschnitt Multi-Factor Authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) die Option Assign MFA device (MFA-Gerät zuweisen).
6. Geben Sie auf der Seite MFA-Gerätename einen Gerätenamen ein, wählen Sie Hauptschlüssel oder Sicherheitsschlüssel und dann Weiter.
7. Richten Sie auf Gerät einrichten Ihren Hauptschlüssel ein. Erstellen Sie einen Hauptschlüssel mit biometrischen Daten wie Ihrem Gesicht oder Fingerabdruck, mit einer Geräte-PIN oder indem Sie den FIDO-Sicherheitsschlüssel in den USB-Anschluss Ihres Computers stecken und darauf tippen.
8. Folgen Sie den Anweisungen in Ihrem Browser und wählen Sie dann Weiter.

Sie haben jetzt einen Hauptschlüssel oder Sicherheitsschlüssel registriert, den ein anderer IAM-Benutzer verwenden kann. AWS Hinweise zur Verwendung von MFA mit dem finden Sie AWS Management Console unterVerwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Ersetzen Sie einen Hauptschlüssel oder Sicherheitsschlüssel

Sie können einem Benutzer mit Ihren und IAM-Benutzern bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen gleichzeitig Root-Benutzer des AWS-Kontos zuweisen. Wenn der Benutzer einen FIDO-Authenticator verliert oder aus anderweitigen Gründen das Gerät ersetzen möchte, müssen Sie zunächst den alten FIDO-Authenticator deaktivieren. Anschließend können Sie ein neues MFA-Gerät für den Benutzer hinzufügen.

- Um ein gegenwärtig mit einem anderen IAM-Benutzer verknüpftes Gerät zu deaktivieren, siehe Deaktivieren von MFA-Geräten.
- Informationen darüber, wie Sie einen neuen FIDO-Sicherheitsschlüssel für einen IAM-Benutzer hinzuzufügen, finden Sie unter Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für Ihren eigenen IAM-Benutzer (Konsole).

Wenn Sie keinen Zugriff auf einen neuen Hauptschlüssel oder Sicherheitsschlüssel haben, können Sie ein neues virtuelles MFA-Gerät oder ein Hardware-TOTP-Token aktivieren. Siehe eine der folgenden Anweisungen:

- Aktivieren eines virtuellen Multi-Factor Authentication (MFA)-Geräts (Konsole)
- Aktivieren eines Hardware-TOTP-Tokens (Konsole)

Unterstützte Konfigurationen für die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln

Sie können gerätegebundene FIDO2-Hauptschlüssel, auch Sicherheitsschlüssel genannt, als Multi-Faktor-Authentifizierungsmethode (MFA) mit IAM verwenden, indem Sie die derzeit unterstützten Konfigurationen verwenden. Dazu gehören FIDO2-Geräte, die von IAM unterstützt werden, und Browser, die FIDO2 unterstützen. Bevor Sie Ihr FIDO2-Gerät registrieren, überprüfen Sie, ob Sie die neueste Browser- und Betriebssystemversion (OS) verwenden. Funktionen können sich in verschiedenen Browsern, Authentifikatoren und Betriebssystemclients unterschiedlich verhalten. Wenn Ihre Geräteregistrierung in einem Browser fehlschlägt, können Sie versuchen, sich mit einem anderen Browser zu registrieren.

FIDO2 ist ein offener Authentifizierungsstandard und eine Erweiterung von FIDO U2F, der das gleiche hohe Sicherheitsniveau auf der Grundlage der Public-Key-Kryptografie bietet. FIDO2 besteht

aus der W3C Web Authentication Specification (WebAuthn API) und dem FIDO Alliance Client-to-Authenticator Protocol (CTAP), einem Protokoll auf Anwendungsebene. CTAP ermöglicht die Kommunikation zwischen Client oder Plattform, wie einem Browser oder Betriebssystem, mit einem externen Authenticator. Wenn Sie einen FIDO-zertifizierten Authentifikator aktivieren AWS, erstellt der Sicherheitsschlüssel ein neues key pair, das nur mit verwendet werden kann. AWS Geben Sie zuerst Ihre Anmeldeinformationen ein. Wenn Sie dazu aufgefordert werden, tippen Sie auf den Sicherheitsschlüssel, der auf die Authentifizierungsherausforderung von reagiert. AWS Weitere Informationen zum FIDO2-Standard finden Sie unter FIDO2-Projekt.

Von AWS unterstützte FIDO2-Geräte

IAM unterstützt FIDO2-Sicherheitsgeräte, die über USB, Bluetooth oder NFC eine Verbindung zu Ihren Geräten herstellen. IAM unterstützt auch Plattformauthentifikatoren wie TouchID, FaceID oder Windows Hello.

> **Note**
>
> AWS benötigt Zugriff auf den physischen USB-Anschluss Ihres Computers, um Ihr FIDO2-Gerät zu verifizieren. Sicherheitsschlüssel funktionieren nicht mit einer virtuellen Maschine, einer Remoteverbindung oder dem Inkognito-Modus eines Browsers.

Die FIDO Alliance führt eine Liste aller FIDO2-Produkte, die mit den FIDO-Spezifikationen kompatibel sind.

Browser, die FIDO2 unterstützen

Die Verfügbarkeit von FIDO2-Sicherheitsgeräten, die in einem Webbrowser ausgeführt werden, hängt von der Kombination aus Browser und Betriebssystem ab. Die folgenden Browser unterstützen derzeit die Verwendung von Sicherheitsschlüsseln:

	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Ja	Ja	Ja	Ja	Nein
Safari	Ja	Nein	Nein	Ja	Nein
Edge	Ja	Ja	Nein	Ja	Nein

	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Firefox	Ja	Ja	Nein	Ja	Nein

> **Note**
>
> Die meisten Firefox-Versionen, die derzeit FIDO2 unterstützen, aktivieren die Unterstützung standardmäßig nicht. Anweisungen zur Aktivierung der FIDO2-Unterstützung in Firefox finden Sie unter. Fehlerbehebung von FIDO-Sicherheitsschlüsseln

Weitere Informationen zur Browserunterstützung für ein FIDO2-zertifiziertes Gerät wie YubiKey finden Sie unter Betriebssystem- und Webbrowser-Unterstützung für FIDO2 und U2F.

Browser-Plugins

AWS unterstützt nur Browser, die FIDO2 nativ unterstützen. AWS unterstützt nicht die Verwendung von Plugins zum Hinzufügen von FIDO2-Browserunterstützung. Einige Browser-Plugins sind nicht mit dem FIDO2-Standard kompatibel und können mit FIDO2-Sicherheitsschlüsseln zu unerwarteten Ergebnissen führen.

Weitere Informationen zum Deaktivieren von Browser-Plugins und andere Tipps zur Fehlerbehebung finden Sie unter Ich kann meinen FIDO-Sicherheitsschlüssel nicht aktivieren.

Gerätezertifizierungen

Gerätebezogene Zertifizierungen wie FIPS-Validierung und FIDO-Zertifizierungsstufe werden nur bei der Registrierung eines Sicherheitsschlüssels erfasst und zugewiesen. Ihre Gerätezertifizierung wird vom FIDO Alliance Metadata Service (MDS) abgerufen. Wenn sich der Zertifizierungsstatus oder die Zertifizierungsstufe Ihres Sicherheitsschlüssels ändert, wird dies nicht automatisch in den Geräte-Tags wiedergegeben. Um die Zertifizierungsinformationen eines Geräts zu aktualisieren, registrieren Sie das Gerät erneut, um die aktualisierten Zertifizierungsinformationen abzurufen.

AWS stellt bei der Geräteregistrierung die folgenden Zertifizierungstypen als Bedingungsschlüssel bereit, die aus dem FIDO MDS abgerufen werden: FIPS-140-2, FIPS-140-3 und FIDO-Zertifizierungsstufen. Sie haben die Möglichkeit, die Registrierung bestimmter Authentifikatoren in ihren IAM-Richtlinien festzulegen, basierend auf Ihrem bevorzugten Zertifizierungstyp und Ihrer bevorzugten Zertifizierungsstufe. Weitere Informationen finden Sie unten unter „Richtlinien“.

Beispielrichtlinien für Gerätezertifizierungen

Die folgenden Anwendungsfälle zeigen Beispielrichtlinien, mit denen Sie MFA-Geräte mit FIPS-Zertifizierungen registrieren können.

Themen

Anwendungsfall 1: Nur die Registrierung von Geräten zulassen, die über eine FIPS-140-2-L2-Zertifizierung verfügen

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
```

```
            }
        }
    ]
}
```

Anwendungsfall 2: Registrierung von Geräten zulassen, die über eine FIPS-140-2-L2- und FIDO-L1-Zertifizierung verfügen

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
}
```

Anwendungsfall 3: Registrierung von Geräten zulassen, die entweder über eine FIPS-140-2-L2- oder eine FIPS-140-3-L2-Zertifizierung verfügen

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
    ]
}
```

Anwendungsfall 4: Ermöglichen Sie die Registrierung von Geräten, die über eine FIPS-140-2-L2-Zertifizierung verfügen und andere MFA-Typen wie virtuelle Authentifikatoren und Hardware-TOTP unterstützen

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Activate",
                    "iam:FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "Null": {
                    "iam:RegisterSecurityKey": "true"
                }
            }
        }
    ]
}
```

AWS CLI und AWS API

AWS unterstützt die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln nur in der AWS Management Console. Die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln für MFA wird in der AWS AND-API oder für den AWS CLIZugriff auf MFA-geschützte API-Operationen nicht unterstützt.

Weitere Ressourcen

- Weitere Informationen zur Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln in finden Sie unter. AWSAktivieren eines Hauptschlüssels oder Sicherheitsschlüssels (Konsole)

- Hilfe zur Problembehebung von Hauptschlüsseln und Sicherheitsschlüsseln in finden Sie AWS unter. Fehlerbehebung von FIDO-Sicherheitsschlüsseln
- Allgemeine Brancheninformationen zur FIDO2-Unterstützung finden Sie unter FIDO2 Project.

Aktivieren eines virtuellen Multi-Factor Authentication (MFA)-Geräts (Konsole)

Sie können ein Telefon oder ein anderes Gerät als virtuelles Multi-Factor-Authentication-(MFA)Gerät verwenden. Installieren Sie dazu eine mobile App mit RFC 6238, einem standardbasierten TOTP-(Time-based One-time Password)Algorithmus. Diese Apps generieren einen sechsstelligen Authentifizierungscode. Da diese Apps auf ungesicherten mobilen Geräten ausgeführt werden können, bietet die virtuelle MFA ggf. nicht denselben Grad an Sicherheit wie FIDO-Sicherheitsschlüssel. Wir empfehlen die Verwendung eines virtuellen MFA-Geräts beim Warten auf die Genehmigung für den Hardware-Kauf oder während Sie warten, bis Ihre Hardware eintrifft.

Die meisten virtuellen MFA-Apps unterstützen die Erstellung mehrerer virtueller Geräte, sodass Sie dieselbe App für mehrere AWS-Konten oder mehrere Benutzer verwenden können. Sie können bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen bei Ihren Root-Benutzer des AWS-Kontos und IAM-Benutzern registrieren. Bei mehreren MFA-Geräten benötigen Sie nur ein MFA-Gerät, um sich AWS CLI als dieser Benutzer anzumelden AWS Management Console oder eine Sitzung über den zu erstellen. Wir empfehlen, mehrere MFA-Geräte zu registrieren. Für Authentifizierungsanwendungen empfehlen wir außerdem, die Cloud-Backup- oder Synchronisierungsfunktion in diesen Apps zu aktivieren. Dadurch vermeiden Sie, dass Sie den Zugriff auf Ihr Konto verlieren, wenn Sie Ihr Gerät mit den Authentifizierungs-Apps verlieren oder beschädigen.

Eine Liste der virtuellen MFA-Apps, die Sie verwenden können, finden Sie unter Multi-Factor Authentication. AWS erfordert eine virtuelle MFA-App, die ein sechsstelliges einmaliges Passwort generiert.

Themen

- Erforderliche Berechtigungen
- Aktivieren eines virtuellen MFA-Geräts für einen IAM-Benutzer (Konsole)
- Ersetzen eines virtuellen MFA-Geräts

Erforderliche Berechtigungen

Um virtuelle MFA-Geräte für Ihren IAM-Benutzer zu verwalten, benötigen Sie die Berechtigungen von der folgenden Richtlinie: AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihr eigenes MFA-Gerät auf der Seite mit den Sicherheitsanmeldedaten zu verwalten.

Aktivieren eines virtuellen MFA-Geräts für einen IAM-Benutzer (Konsole)

Sie können IAM in verwenden, AWS Management Console um ein virtuelles MFA-Gerät für einen IAM-Benutzer in Ihrem Konto zu aktivieren und zu verwalten. Sie können Ihren IAM-Ressourcen, einschließlich virtueller MFA-Geräte, Tags hinzufügen, um den Zugriff auf diese zu identifizieren, zu organisieren und zu kontrollieren. Sie können virtuelle MFA-Geräte nur kennzeichnen, wenn Sie die AWS API AWS CLI oder verwenden. Informationen zum Aktivieren und Verwalten eines MFA-Geräts mithilfe der AWS API AWS CLI oder finden Sie unterAktivierung und Verwaltung virtueller MFA-Geräte (AWS CLI oder AWS API). Weitere Informationen über das Markieren von IAM-Ressourcen mit Tags finden Sie unter Markieren von IAM-Ressourcen.

> **Note**
>
> Sie müssen über physischen Zugriff auf die Hardware verfügen, die als Host für das virtuelle MFA-Gerät des Benutzers dient, um MFA konfigurieren zu können. Beispielsweise können Sie MFA für einen Benutzer konfigurieren, der ein virtuelles MFA-Gerät verwendet, das auf einem Smartphone ausgeführt wird. In diesem Fall müssen Sie das Smartphone zur Verfügung haben, um den Assistenten zu beenden. Aus diesem Grund kann es sinnvoll sein, die Konfiguration und Verwaltung der virtuellen MFA-Geräte von den Benutzern selbst vornehmen zu lassen. In diesem Fall müssen Sie den Benutzern die Berechtigungen zur Ausführung der erforderlichen IAM-Aktionen erteilen. Weitere Informationen und ein Beispiel für eine IAM-Richtlinie, die diese Berechtigungen gewährt, finden Sie unter IAM-Tutorial: Zulassen, dass Ihre Benutzer ihre eigenen Anmeldeinformationen und MFA-Einstellungen konfigurieren können und Beispielrichtlinie AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihr eigenes MFA-Gerät auf der Seite mit den Sicherheitsanmeldedaten zu verwalten.

Aktivieren eines virtuellen MFA-Geräts für einen IAM-Benutzer (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 auf Users (Benutzer).

3. Wählen Sie in der Users list (Liste der Benutzer) den Namen des IAM-Benutzer aus.
4. Wechseln Sie zur Registerkarte Security Credentials. Wählen Sie im Abschnitt Multi-Factor Authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) die Option Assign MFA device (MFA-Gerät zuweisen).
5. Geben Sie im Assistenten einen Device name (Gerätenamen) ein, wählen Sie Authenticator app (Authenticator-App) und dann Next (Weiter).

   IAM generiert Konfigurationsinformationen für das virtuelle MFA-Gerät und zeigt diese einschließlich eines QR-Codes an. Dieser Code ist eine grafische Darstellung des "geheimen Konfigurationsschlüssels", der für die manuelle Eingabe auf Geräte zur Verfügung steht, die keine QR-Codes unterstützen.
6. Öffnen Sie Ihre virtuelle MFA-App. Eine Liste der Anwendungen, die Sie zum Hosten von virtuellen MFA-Geräten verwenden können, finden Sie unter Multi-Factor Authentication.

   Wenn die virtuelle MFA-App mehrere virtuelle MFA-Geräte oder -Konten unterstützt, wählen Sie die Option zum Erstellen eines neuen virtuellen MFA-Geräts oder -Kontos.
7. Stellen Sie fest, ob die MFA-App QR-Codes unterstützt, und führen Sie dann einen der folgenden Schritte aus:

   - Wählen Sie im Assistenten Show QR code (QR-Code anzeigen) und verwenden Sie dann die App, um den QR-Code zu scannen. Sie können beispielsweise das Kamerasymbol oder eine Anwendung wie z. B. Scan Code (Code scannen) auswählen und dann mit der Kamera des Geräts den Code scannen.
   - Wählen Sie im Assistenten die Option Show secret key (Geheimen Schlüssel anzeigen) aus und geben Sie dann den geheimen Schlüssel in Ihre MFA-App ein.

   Wenn Sie fertig sind, beginnt das virtuelle MFA-Gerät, einmalige Passwörter zu generieren.
8. Geben Sie auf der Seite Set up device (Gerät einrichten) im Feld MFA code 1 (MFA-Code 1) das aktuell auf dem virtuellen MFA-Gerät angezeigte Einmalpasswort ein. Warten Sie bis zu 30 Sekunden, bis das Gerät ein neues einmaliges Passwort generiert. Geben Sie dann das zweite Einmalpasswort in das Feld MFA Code 2 ein. Wählen Sie Add MFA (MFA hinzufügen).

   > **Important**
   >
   > Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes generieren und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit

dem Benutzer verknüpft, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. In diesem Fall können Sie das Gerät neu synchronisieren.

Das virtuelle MFA-Gerät ist jetzt für die Verwendung mit AWS bereit. Hinweise zur Verwendung von MFA mit dem finden Sie AWS Management Console unterVerwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Ersetzen eines virtuellen MFA-Geräts

Sie können bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen bei Ihren Root-Benutzer des AWS-Kontos und IAM-Benutzern registrieren. Wenn der Benutzer ein Gerät verliert oder aus anderweitigen Gründen das Gerät ersetzen möchte, müssen Sie zunächst das alte Gerät deaktivieren. Anschließend können Sie das neue Gerät für den Benutzer hinzufügen.

- Um ein gegenwärtig mit einem anderen IAM-Benutzer verknüpftes Gerät zu deaktivieren, siehe Deaktivieren von MFA-Geräten.
- Um ein virtuelles MFA-Ersatzgerät für einen anderen IAM-Benutzer hinzuzufügen, befolgen Sie die Anleitung Aktivieren eines virtuellen MFA-Geräts für einen IAM-Benutzer (Konsole) oben.
- Gehen Sie wie im Verfahren Aktivieren eines virtuellen MFA-Geräts für Ihren Root-Benutzer des AWS-Kontos (Konsole) beschrieben vor Root-Benutzer des AWS-Kontos, um ein virtuelles MFA-Ersatzgerät für das hinzuzufügen.

Aktivieren eines Hardware-TOTP-Tokens (Konsole)

Ein Hardware-TOTP-Token generiert auf der Grundlage des Algorithmus für ein zeitgesteuertes Einmalpasswort (TOTP) einen sechsstelligen numerischen Code. Der Benutzer muss einen gültigen Code vom Gerät eingeben, wenn er während des Anmeldevorgangs dazu aufgefordert wird. Ein MFA-Gerät muss einem Benutzer eindeutig zugewiesen werden. Ein Benutzer kann sich nicht mit dem Code eines anderen Geräts authentifizieren. MFA-Geräte können nicht von Konten oder Benutzern gemeinsam genutzt werden.

Hardware-TOTP-Token und FIDO-Sicherheitsschlüssel sind physische Geräte, die Sie kaufen können. Hardware-MFA-Geräte generieren TOTP-Codes für die Authentifizierung, wenn Sie sich anmelden. AWS Sie sind auf Batterien angewiesen, die im Laufe der Zeit möglicherweise ausgetauscht und neu synchronisiert werden müssen. AWS FIDO-Sicherheitsschlüssel, die Kryptografie mit öffentlichen Schlüsseln verwenden, benötigen keine Batterien und bieten

einen nahtlosen Authentifizierungsprozess. Wir empfehlen die Verwendung von FIDO-Sicherheitsschlüsseln aufgrund ihrer Phishing-Resistenz, die eine sicherere Alternative zu TOTP-Geräten darstellt. Darüber hinaus können FIDO-Sicherheitsschlüssel mehrere IAM- oder Root-Benutzer auf demselben Gerät unterstützen, wodurch ihr Nutzen für die Kontosicherheit verbessert wird. Spezifikationen und Kaufinformationen für beide Gerätetypen finden Sie unter Multifaktor-Authentifizierung.

Sie können ein Hardware-TOTP-Token für einen IAM-Benutzer über die AWS Management Console Befehlszeile oder die IAM-API aktivieren. Informationen zum Aktivieren eines MFA-Geräts für Sie finden Sie Root-Benutzer des AWS-Kontos unterAktivieren Sie ein Hardware-TOTP-Token für die Root-Benutzer des AWS-Kontos (Konsole).

Sie können bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen bei Ihren Root-Benutzer des AWS-Kontos und IAM-Benutzern registrieren. Bei mehreren MFA-Geräten benötigen Sie nur ein MFA-Gerät, um sich AWS CLI als dieser Benutzer anzumelden AWS Management Console oder eine Sitzung über den zu erstellen.

> **Important**
>
> Wir empfehlen Ihnen, mehrere MFA-Geräte für Ihre Benutzer zu aktivieren, damit Sie im Falle eines verlorenen oder unzugänglichen MFA-Geräts weiterhin auf Ihr Konto zugreifen können.

> **Note**
>
> Wenn Sie das Gerät über die Befehlszeile aktivieren möchten, verwenden Sie `aws iam enable-mfa-device`. Um das MFA-Gerät mit der IAM-API zu aktivieren, verwenden Sie die Operation `EnableMFADevice`.

Themen

Erforderliche Berechtigungen

Um ein physisches TOTP-Token für Ihren eigenen IAM-Benutzer zu verwalten und dabei gleichzeitig sensible MFA-bezogene Aktionen zu schützen, benötigen Sie die Berechtigungen von der folgenden Richtlinie:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Aktivieren eines physischen TOTP-Tokens für Ihren eigenen IAM-Benutzer (Konsole)

Sie können Ihr eigenes physisches TOTP-Token über die AWS Management Console aktivieren.

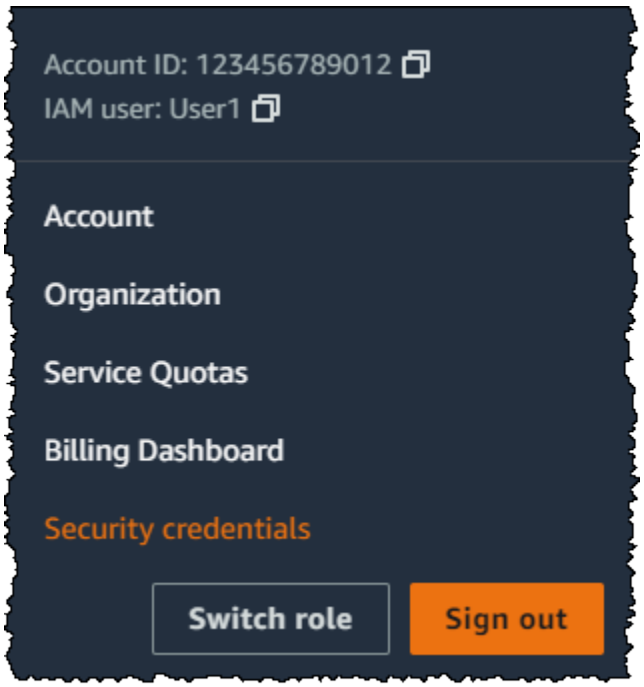
> **Note**
>
> Bevor Sie ein physisches TOTP-Token aktivieren können, benötigen Sie physischen Zugriff auf das Gerät.

So aktivieren Sie ein physisches TOTP-Token für Ihren eigenen IAM-Benutzer (Konsole)

1. Verwenden Sie Ihre AWS Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der IAM-Konsole anzumelden.

   > **Note**
   >
   > Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich Ihren IAM-Benutzernamen und Ihre Kontoinformationen zu merken. Wenn Sie sich zuvor als anderer Benutzer angemeldet haben, wählen Sie Melden Sie sich bei einem anderen Konto an Um zur Hauptanmeldeseite zurückzukehren. Von dort aus können Sie Ihre AWS Konto-ID oder Ihren Kontoalias eingeben, um zur IAM-Benutzer-Anmeldeseite für Ihr Konto weitergeleitet zu werden.

   Wenden Sie sich an Ihren Administrator, um Ihre AWS-Konto ID zu erhalten.
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheitsanmeldeinformationen) aus.



3. Wählen Sie auf der Registerkarte AWS -IAM-Anmeldeinformationen im Abschnitt Multi-Faktor-Authentifizierung (MFA) die Option MFA-Gerät zuweisen.
4. Geben Sie im Assistenten einen Device name (Gerätenamen) ein, wählen Sie Hardware TOTP token (Physisches TOTP-Token) und dann Next (Weiter).
5. Geben Sie die Seriennummer des Geräts ein. Die Seriennummer befindet sich in der Regel auf der Rückseite des Geräts.
6. Geben Sie im Feld MFA code 1 (MFA-Code 1) die am MFA-Gerät angezeigte sechsstellige Nummer ein. Sie müssen eventuell die Taste auf der Vorderseite des Geräts drücken, um die Zahl anzuzeigen.

7. Warten Sie 30 Sekunden, während das Gerät den Code aktualisiert, und geben Sie dann die nächste sechsstellige Nummer in das Feld MFA code 2 (MFA-Code 2) ein. Sie müssen eventuell die Taste auf der Vorderseite des Geräts drücken, um die zweite Zahl anzuzeigen.
8. Wählen Sie Add MFA (MFA hinzufügen).

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Authentifizierungscodes. Wenn Sie die Codes erzeugen und zu lange mit der Anforderung warten, wird das MFA-

> Gerät erfolgreich mit dem Benutzer verknüpft, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. In diesem Fall können Sie das Gerät neu synchronisieren.

Das Gerät ist bereit für die Verwendung mit AWS. Weitere Informationen zur Verwendung von MFA mit der AWS Management Console finden Sie unter Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Aktivieren eines physischen TOTP-Tokens für einen anderen IAM-Benutzer (Konsole)

Sie können ein physisches TOTP-Token über die AWS Management Console für einen anderen IAM-Benutzer aktivieren.

So aktivieren Sie ein physisches TOTP-Token für einen anderen IAM-Benutzer (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 auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers, für den Sie MFA aktivieren möchten.
4. Wechseln Sie zur Registerkarte Security Credentials. Wählen Sie im Abschnitt Multi-Factor Authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) die Option Assign MFA device (MFA-Gerät zuweisen).
5. Geben Sie im Assistenten einen Device name (Gerätenamen) ein, wählen Sie Hardware TOTP token (Physisches TOTP-Token) und dann Next (Weiter).
6. Geben Sie die Seriennummer des Geräts ein. Die Seriennummer befindet sich in der Regel auf der Rückseite des Geräts.
7. Geben Sie im Feld MFA code 1 (MFA-Code 1) die am MFA-Gerät angezeigte sechsstellige Nummer ein. Sie müssen eventuell die Taste auf der Vorderseite des Geräts drücken, um die Zahl anzuzeigen.

8. Warten Sie 30 Sekunden, während das Gerät den Code aktualisiert, und geben Sie dann die nächste sechsstellige Nummer in das Feld MFA code 2 (MFA-Code 2) ein. Sie müssen eventuell die Taste auf der Vorderseite des Geräts drücken, um die zweite Zahl anzuzeigen.

9. Wählen Sie Add MFA (MFA hinzufügen).

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Authentifizierungscodes. Wenn Sie die Codes erzeugen und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit dem Benutzer verknüpft, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. In diesem Fall können Sie das Gerät neu synchronisieren.

Das Gerät ist bereit für die Verwendung mit AWS. Weitere Informationen zur Verwendung von MFA mit der AWS Management Console finden Sie unter Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Ersetzen eines physischen MFA-Geräts

Sie können einem Benutzer mit Ihren und IAM-Benutzern bis zu acht MFA-Geräte mit einer beliebigen Kombination der derzeit unterstützten MFA-Typen gleichzeitig Root-Benutzer des AWS-Kontos zuweisen. Wenn der Benutzer ein Gerät verliert oder aus anderweitigen Gründen das Gerät ersetzen möchte, müssen Sie zunächst das alte Gerät deaktivieren. Anschließend können Sie das neue Gerät für den Benutzer hinzufügen.

- Um ein gegenwärtig mit einem Benutzer verknüpftes Gerät zu deaktivieren, siehe Deaktivieren von MFA-Geräten.
- Um ein physisches TOTP-Ersatztoken für einen IAM-Benutzer hinzuzufügen, führen Sie die in der Anleitung Aktivieren eines physischen TOTP-Tokens für einen anderen IAM-Benutzer (Konsole) weiter oben in diesem Thema beschriebenen Schritte.
- Gehen Sie wie im Verfahren weiter oben in diesem Thema beschrieben vor Root-Benutzer des AWS-Kontos, um ein Ersatz-Hardware-TOTP-Token für das hinzufügenAktivieren Sie ein Hardware-TOTP-Token für die Root-Benutzer des AWS-Kontos (Konsole).

Aktivierung und Verwaltung virtueller MFA-Geräte (AWS CLI oder AWS API)

Sie können AWS CLI Befehle oder AWS API-Operationen verwenden, um ein virtuelles MFA-Gerät für einen IAM-Benutzer zu aktivieren. Sie können ein MFA-Gerät nicht Root-Benutzer des AWS-Kontos mit der AWS API AWS CLI, Tools für Windows PowerShell oder einem anderen

Befehlszeilentool für aktivieren. Sie können jedoch das verwenden, AWS Management Console um ein MFA-Gerät für den Root-Benutzer zu aktivieren.

Wenn Sie ein MFA-Gerät von der aus aktivieren AWS Management Console, führt die Konsole mehrere Schritte für Sie aus. Wenn Sie stattdessen ein virtuelles Gerät mithilfe von Tools für Windows PowerShell oder AWS API erstellen, müssen Sie die Schritte manuell und in der richtigen Reihenfolge ausführen. AWS CLI Um beispielsweise ein virtuelles MFA-Gerät zu erstellen, müssen Sie das IAM-Objekt erstellen und den Code entweder als Zeichenfolge oder QR-Code extrahieren. Anschließend müssen Sie das Gerät synchronisieren und einem IAM-Benutzer zuordnen. Weitere Informationen finden Sie unter New-IAMVirtualMFADevice im Abschnitt Examples (Beispiele). Bei physischen Geräten können Sie den Erstellungsschritt überspringen und das Gerät direkt synchronisieren und dem Benutzer zuordnen.

Sie können Ihren IAM-Ressourcen, einschließlich virtueller MFA-Geräte, Tags hinzufügen, um den Zugriff auf diese zu identifizieren, zu organisieren und zu kontrollieren. Sie können virtuelle MFA-Geräte nur kennzeichnen, wenn Sie die AWS API AWS CLI oder verwenden.

Ein IAM-Benutzer, der das SDK oder die CLI verwendet, kann ein zusätzliches MFA-Gerät aktivieren, indem er `EnableMFADevice` aufruft oder ein vorhandenes MFA-Gerät durch den Aufruf von `DeactivateMFADevice` deaktivieren. Um dies erfolgreich durchzuführen, müssen sie zuerst `GetSessionToken` aufrufen und MFA-Codes mit einem vorhandenen MFA-Gerät senden. Dieser Aufruf gibt temporäre Anmeldeinformationen zurück, die dann zum Signieren von API-Vorgängen verwendet werden können, die eine MFA-Authentifizierung erfordern. Ein Beispiel für Anforderung und Antwort finden Sie unter `GetSessionToken` – Temporäre Anmeldeinformationen für Benutzer in nicht vertrauenswürdigen Umgebungen.

So erstellen Sie die virtuelle Geräteeinheit in IAM, um ein virtuelles MFA-Gerät zu repräsentieren

Mithilfe dieser Befehl wird ein ARN für das Gerät bereitgestellt, der in vielen der folgenden Befehle anstelle einer Seriennummer verwendet wird.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API: `CreateVirtualMFADevice`

So aktivieren Sie ein MFA-Gerät für die Verwendung mit AWS

Mit diesen Befehlen wird das Gerät mit einem Benutzer synchronisiert AWS und diesem zugeordnet. Wenn es sich dabei um ein virtuelles Gerät handelt, verwenden Sie den ARN des virtuellen Geräts als Seriennummer.

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Authentifizierungscodes. Wenn Sie die Codes erzeugen und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit dem Benutzer verknüpft, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden. Führen Sie in diesem Fall mithilfe der unten beschriebenen Befehle eine erneute Synchronisierung durch.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API: `EnableMFADevice`

So deaktivieren Sie ein Gerät

Mit diesen Befehlen heben Sie die Zuordnung des Geräts zu einem Benutzer auf und deaktivieren es. Wenn es sich dabei um ein virtuelles Gerät handelt, verwenden Sie den ARN des virtuellen Geräts als Seriennummer. Außerdem müssen Sie die virtuelle Geräteeinheit separat löschen.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API: `DeactivateMFADevice`

So listen Sie virtuelle MFA-Geräteeinheiten auf

Verwenden Sie diese Befehle, um virtuelle MFA-Geräte-Entitys aufzulisten.

- AWS CLI: `aws iam list-virtual-mfa-devices`
- AWS API: `ListVirtualMFADevices`

Markieren eines virtuellen MFA-Geräts

Verwenden Sie diese Befehle, um ein virtuelles MFA-Gerät zu markieren.

- AWS CLI: `aws iam tag-mfa-device`
- AWS API: `TagMFADevice`

Auflisten der Tags für ein virtuelles MFA-Gerät

Verwenden Sie diese Befehle, um die an ein virtuelles MFA-Gerät angehängten Tags aufzulisten.

- AWS CLI: `aws iam list-mfa-device-tags`
- AWS API: `ListMFADeviceTags`

Entfernen einer Markierung eines virtuellen MFA-Geräts

Verwenden Sie diese Befehle, um Tags zu entfernen, die an ein virtuelles MFA-Gerät angehängt sind.

- AWS CLI: `aws iam untag-mfa-device`
- AWS API: `UntagMFADevice`

So führen Sie eine erneute Synchronisierung eines MFA-Geräts durch

Verwenden Sie diese Befehle, wenn das Gerät Codes generiert, die von nicht akzeptiert werden AWS. Wenn es sich dabei um ein virtuelles Gerät handelt, verwenden Sie den ARN des virtuellen Geräts als Seriennummer.

- AWS CLI: `aws iam resync-mfa-device`
- AWS API: `ResyncMFADevice`

So löschen Sie eine virtuelle MFA-Geräteeinheit in IAM

Nachdem Sie die Gerätezuordnung zu einem Benutzer aufgehoben haben, können Sie die Geräteeinheit löschen.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API: `DeleteVirtualMFADevice`

So stellen Sie ein verlorenes oder nicht mehr funktionierendes virtuelles MFA-Gerät wieder her

Manchmal geht ein mobiles Gerät eines Benutzers, auf dem die virtuelle MFA-Anwendung gehostet wird, verloren, wird ausgetauscht oder es funktioniert nicht mehr. Wenn dies der Fall ist, kann der Benutzer es nicht selbst wiederherstellen. Der Benutzer muss sich zur Deaktivierung des Geräts an einen Administrator wenden. Weitere Informationen finden Sie unter Was passiert, wenn ein MFA-Gerät verloren geht oder nicht funktioniert?.

Überprüfen des MFA-Status

Verwenden Sie die IAM-Konsole, um zu überprüfen, ob ein Root-Benutzer des AWS-Kontos oder IAM-Benutzer ein gültiges MFA-Gerät aktiviert hat.

So überprüfen Sie den MFA-Status eines Stammbenutzers

1. Melden Sie sich AWS Management Console mit Ihren Root-Benutzeranmeldedaten bei an und öffnen Sie dann die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheitsanmeldeinformationen) aus.
3. Überprüfen Sie unter Multi-Factor Authentication (MFA), ob MFA aktiviert oder deaktiviert ist. Wenn MFA nicht aktiviert ist, wird ein Warnsymbol () angezeigt.

Wenn Sie MFA für das Konto aktivieren möchten, finden Sie Informationen unter:

- Aktivieren eines virtuellen MFA-Geräts für Ihren Root-Benutzer des AWS-Kontos (Konsole)
- Aktivieren Sie einen Hauptschlüssel oder Sicherheitsschlüssel für die Root-Benutzer des AWS-Kontos (Konsole)
- Aktivieren Sie ein Hardware-TOTP-Token für die Root-Benutzer des AWS-Kontos (Konsole)

So überprüfen Sie den MFA-Status des IAM-Benutzers

1. Öffnen Sie unter https://console.aws.amazon.com/iam/ die IAM-Konsole.
2. Klicken Sie im Navigationsbereich auf Users (Benutzer).
3. Falls erforderlich, fügen Sie die Spalte MFA zur Tabelle "Benutzer" hinzu, indem Sie die folgenden Schritte ausführen:
   a. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ().
   b. In Manage Columns (Spalten verwalten) wählen Sie MFA.
   c. (Optional) Entfernen Sie die Markierung der Kontrollkästchen für alle Spaltenüberschriften, die nicht in der Tabelle Benutzer erscheinen sollen.

d. Klicken Sie auf Close (Schließen), um zur Liste der Benutzer zurückzukehren.

4. In der Spalte MFA erhalten Sie Informationen über das aktivierte MFA-Gerät. Wenn kein MFA-Gerät für den Benutzer aktiv ist, zeigt die Konsole None (Keine) an. Wenn der Benutzer ein aktiviertes MFA-Gerät hat, zeigt die Spalte MFA den Typ des Geräts, das mit einem Wert von Virtual, Sicherheitsschlüssel, Hardware oder SMS aktiviert ist.

   > **Note**
   >
   > AWS Die Unterstützung für die Aktivierung der SMS-Multifaktor-Authentifizierung (MFA) wurde eingestellt. Wir empfehlen Kunden mit IAM-Benutzern, die SMS-textnachrichtenbasierte MFA verwenden, zu einer der folgenden alternativen Methoden zu wechseln: virtuelles (softwarebasiertes) MFA-Gerät, FIDO-Sicherheitsschlüssel, oder Hardware-MFA-Gerät. Sie können die Benutzer in Ihrem Konto mit einem zugewiesenen SMS-MFA-Gerät identifizieren. Navigieren Sie zu diesem Zweck zur IAM-Konsole, wählen Sie im Navigationsbereich Users (Benutzer) und suchen Sie nach Benutzern mit SMS in der Tabellenspalte MFA.

5. Um zusätzliche Informationen über das MFA-Gerät für einen Benutzer anzuzeigen, wählen Sie den Namen des Benutzers, dessen MFA-Status Sie überprüfen möchten. Klicken Sie dann auf die Registerkarte Security credentials (Sicherheits-Anmeldeinformationen).
6. Wenn kein MFA-Gerät für den Benutzer aktiv ist, zeigt die Konsole No MFA devices (Keine MFA-Geräte) an. Weisen Sie im Abschnitt Multi-Factor Authentication (MFA) ein MFA-Gerät zu, um die Sicherheit Ihrer AWS Umgebung zu verbessern. Wenn der Benutzer MFA-Geräte aktiviert hat, werden im Abschnitt Multi-Faktor-Authentifizierung (MFA) Details zu den Geräten angezeigt:

   - Der Gerätename
   - Der Gerätetyp
   - Die Kennung für das Gerät, z. B. eine Seriennummer für ein physisches Gerät oder der ARN AWS für ein virtuelles Gerät
   - Wann das Gerät erstellt wurde

Um ein Gerät zu entfernen oder erneut zu synchronisieren, wählen Sie das Optionsfeld neben dem Gerät und wählen Sie Remove (Entfernen) oder Resync (Erneut synchronisieren).

Weitere Informationen zur Aktivierung von MFA finden Sie unter:

- Aktivieren eines virtuellen Multi-Factor Authentication (MFA)-Geräts (Konsole)

- Aktivieren eines Hauptschlüssels oder Sicherheitsschlüssels (Konsole)
- Aktivieren eines Hardware-TOTP-Tokens (Konsole)

Resynchronisieren von virtuellen und physischen MFA-Geräten

Sie können AWS es verwenden, um Ihre virtuellen und Hardware-Geräte mit Multi-Faktor-Authentifizierung (MFA) neu zu synchronisieren. Wenn das Gerät nicht synchronisiert ist, wenn Sie versuchen, es zu verwenden, schlägt der Anmeldeversuch fehl, und IAM fordert Sie auf, das Gerät erneut zu synchronisieren.

> **Note**
>
> FIDO-Sicherheitsschlüssel verlieren ihre Synchronisation nicht. Wenn ein FIDO-Sicherheitsschlüssel verloren geht oder beschädigt ist, können Sie ihn deaktivieren. Weitere Anweisungen zum Deaktivieren von MFA-Geräten finden Sie unter So deaktivieren Sie ein MFA-Gerät für einen anderen IAM-Benutzer (Konsole).

Als AWS Administrator können Sie die virtuellen und Hardware-MFA-Geräte Ihrer IAM-Benutzer erneut synchronisieren, wenn sie nicht mehr synchronisiert sind.

Wenn Ihr Root-Benutzer des AWS-Kontos MFA-Gerät nicht funktioniert, können Sie Ihr Gerät mithilfe der IAM-Konsole mit oder ohne Abschluss des Anmeldevorgangs neu synchronisieren. Wenn Sie Ihr Gerät nicht erfolgreich resynchronisieren können, müssen Sie es möglicherweise trennen und erneut verknüpfen. Weitere Information dazu finden Sie unter Deaktivieren von MFA-Geräten und Aktivierung von MFA-Geräten für Benutzer in AWS.

Themen

- Erforderliche Berechtigungen
- Erneutes Synchronisieren virtueller und physischer MFA-Geräte (IAM-Konsole)
- Resynchronisieren von virtuellen und physischen MFA-Geräten (AWS CLI)
- Resynchronisierung von virtuellen und Hardware-MFA-Geräten (API)AWS

Erforderliche Berechtigungen

Um virtuelle oder Hardware-MFA-Geräte für Ihren IAM-Benutzer erneut zu synchronisieren, benötigen Sie die Berechtigungen von der folgenden Richtlinie. Diese Richtlinie erlaubt es Ihnen nicht, ein Gerät zu erstellen oder zu deaktivieren.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Erneutes Synchronisieren virtueller und physischer MFA-Geräte (IAM-Konsole)

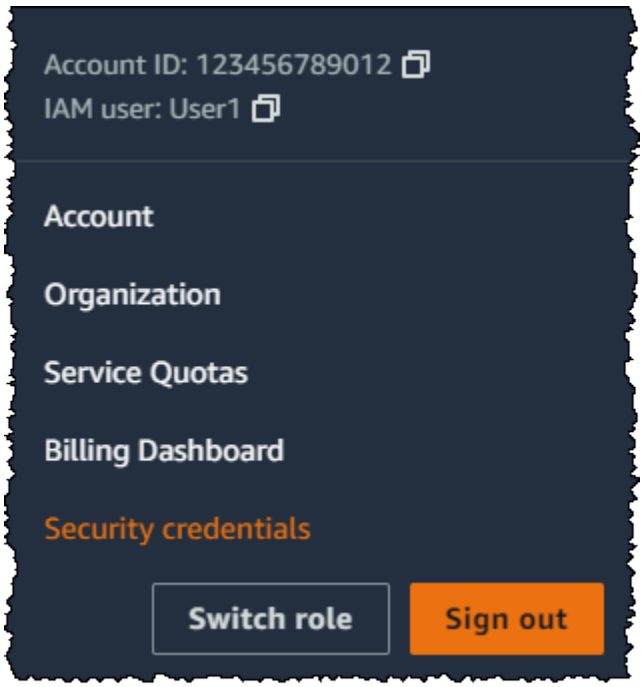
Sie können die IAM Konsole verwenden, um eine Neusynchronisierung bei physischen und virtuellen MFA-Geräten durchzuführen.

So synchronisieren Sie ein virtuelles oder physisches MFA-Gerät für Ihren eigenen IAM-Benutzer (Konsole) neu

1. Verwenden Sie Ihre AWS Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der IAM-Konsole anzumelden.

   > **Note**
   >
   > Der Einfachheit halber verwendet die AWS Anmeldeseite ein Browser-Cookie, um sich Ihren IAM-Benutzernamen und Ihre Kontoinformationen zu merken. Wenn Sie sich zuvor als anderer Benutzer angemeldet haben, wählen Sie Melden Sie sich bei einem anderen Konto an Um zur Hauptanmeldeseite zurückzukehren. Von dort aus können Sie Ihre AWS Konto-ID oder Ihren Kontoalias eingeben, um zur IAM-Benutzer-Anmeldeseite für Ihr Konto weitergeleitet zu werden.

   Wenden Sie sich an Ihren Administrator, um Ihre AWS-Konto ID zu erhalten.
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheitsanmeldeinformationen) aus.



3. Wählen Sie auf der Registerkarte AWS -IAM-Anmeldeinformationen im Abschnitt Multi-Faktor-Authentifizierung (MFA) das Optionsfeld neben dem MFA-Gerät und dann Erneut synchronisieren.
4. Geben Sie unter MFA code 1 (MFA-Code 1) und MFA code 2 (MFA-Code 2) die nächsten zwei sequenziell generierten Codes des Geräts ein. Wählen Sie dann Resync (Erneut synchronisieren).

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes erzeugen und zu lange mit der Anforderung warten, scheint die Anforderung zu funktionieren, aber das Gerät ist weiterhin nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden.

So synchronisieren Sie ein virtuelles oder physisches MFA-Gerät für einen anderen IAM-Benutzer (Konsole) neu

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 Users (Benutzer) den Namen des Benutzers aus, dessen MFA-Gerät erneut synchronisiert werden muss.

3. Wechseln Sie zur Registerkarte Sicherheitsanmeldeinformationen. Wählen Sie im Abschnitt Multi-factor authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) das Optionsfeld neben dem MFA-Gerät und dann Resync (Erneut synchronisieren).
4. Geben Sie unter MFA code 1 (MFA-Code 1) und MFA code 2 (MFA-Code 2) die nächsten zwei sequenziell generierten Codes des Geräts ein. Wählen Sie dann Resync (Erneut synchronisieren).

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes erzeugen und zu lange mit der Anforderung warten, scheint die Anforderung zu funktionieren, aber das Gerät ist weiterhin nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden.

So synchronisieren Sie Ihr Stammbenutzer-MFA vor der Anmeldung erneut (Konsole)

1. Wählen Sie auf der Seite Amazon Web Services Sign In With Authentication Device (Amazon Web Services Anmeldung mit Authentifizierungsgerät) die Option Having problems with your authentication device? (Haben Sie Probleme mit Ihrem Authentifizierungsgerät?) Click here.

> **Note**
>
> Möglicherweise wird Ihnen unterschiedlicher Text angezeigt, z. B. Sign in using MFA (Melden Sie sich mit MFA an) und Troubleshoot your authentication device (Fehlerbehebung bei Ihrem Authentifizierungsgerät). Es stehen jedoch die gleichen Features zur Verfügung.

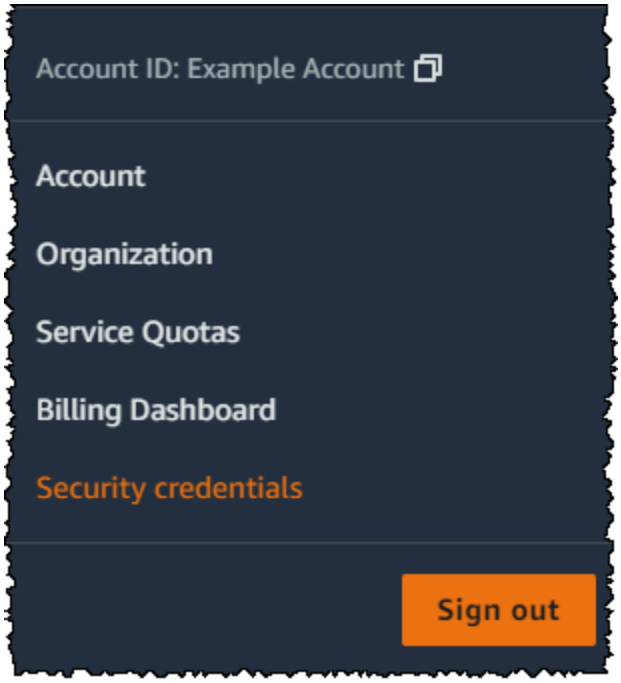
2. Geben Sie im Abschnitt Re-Sync With Our Servers (Re-Synchronisation mit unseren Servern) unter MFA code 1 (MFA-Code 1) und MFA code 2 (MFA-Code 2) die beiden nächsten in der Folge generierten Codes von dem Gerät ein. Wählen Sie dann Re-sync authentication device (Authentifizierungsgerät neu synchronisieren).
3. Geben Sie, falls erforderlich, das Passwort erneut ein, und wählen Sie Sign in (Anmelden). Schließen Sie dann mit Ihrem MFA-Gerät den Anmeldevorgang ab.

So synchronisieren Sie Ihr Stammbenutzer-MFA-Gerät nach der Anmeldung erneut (Konsole)

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

   > **Note**
   >
   > Als Root-Benutzer können Sie sich nicht auf der Seite Sign in as IAM user (Als IAM-Benutzer anmelden) anmelden. Wenn Ihnen die Seite Sign in as IAM user (Als IAM-Benutzer anmelden) angezeigt wird, wählen Sie die Option Sign in using root user email (Mit Root-Benutzer-E-Mail anmelden) unten auf der Seite. Hilfe bei der Anmeldung als Root-Benutzer finden Sie im Benutzerhandbuch unter AWS Management Console Als AWS-Anmeldung Root-Benutzer anmelden.

2. Klicken Sie rechts in der Navigationsleiste auf den Kontonamen und wählen Sie dann Security Credentials (Sicherheitsanmeldeinformationen). Sofern erforderlich, wählen Sie Continue to Security Credentials (Weiter zu Sicherheitsanmeldeinformationen).



3. Erweitern Sie den Bereich Multi-Factor Authentication (MFA) auf der Seite.
4. Aktivieren Sie das Optionsfeld neben dem Gerät und wählen Sie Resync (Erneut synchronisieren).

5. Geben Sie im Dialogfeld Resync MFA device (MFA-Gerät erneut synchronisieren) die beiden nächsten in der Folge generierten Codes vom Gerät unter MFA code 1 (MFA-Code 1) und MFA code 2 (MFA-Code 2) ein. Wählen Sie dann Resync (Erneut synchronisieren).

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes generieren und zu lange mit der Anforderung warten, wird das MFA-Gerät erfolgreich mit dem Benutzer verbunden, aber das Gerät ist nicht synchronisiert. Dies liegt daran, weil die zeitgesteuerten Einmalpasswörter (TOTP) nach einer kurzen Zeit ungültig werden.

Resynchronisieren von virtuellen und physischen MFA-Geräten (AWS CLI)

Sie können die AWS CLI verwenden, um eine Neusynchronisierung bei physischen und virtuellen MFA-Geräten durchzuführen.

So synchronisieren Sie ein virtuelles oder Hardware-MFA-Gerät für einen IAM-Benutzer neu (AWS CLI)

Geben Sie an der Befehlszeile den Befehl aws iam resync-mfa-device ein:

- Virtuelles MFA-Gerät: Geben Sie den Amazon Resource Name (ARN) des Geräts als Seriennummer ein.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Physisches MFA-Gerät: Geben Sie die Seriennummer des physischen Geräts als Seriennummer an. Das Format ist herstellerspezifisch. Sie können beispielsweise ein Gemalto-Token von Amazon erwerben. Die Seriennummer besteht in der Regel aus vier Buchstaben, gefolgt von vier Ziffern.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Senden Sie die Anforderung direkt nach der Erzeugung der Codes. Wenn Sie die Codes generieren und zu lange mit dem Senden der Anforderung warten, schlägt die Anforderung fehl, weil die Gültigkeit der Codes nach kurzer Zeit verfällt.

Resynchronisierung von virtuellen und Hardware-MFA-Geräten (API)AWS

IAM bietet einen API-Aufruf zur Durchführung der Synchronisierung. In diesem Fall empfehlen wir, dass Sie Ihren Benutzern für virtuelle und Hardware-MFA-Geräte die Berechtigung erteilen, auf diesen API-Aufruf zuzugreifen. Erstellen Sie dann ein Tool basierend auf diesem API-Aufruf, damit Ihre Benutzer ihre Geräte jederzeit neu synchronisieren können.

So synchronisieren Sie ein virtuelles oder Hardware-MFA-Gerät für einen IAM-Benutzer (API) neu AWS

- Senden Sie die ResyncMFADevice-Anfrage.

Deaktivieren von MFA-Geräten

Wenn Sie Schwierigkeiten haben, sich mit einem Multi-Faktor-Authentifizierung (MFA)-Gerät als IAM-Benutzer anzumelden, wenden Sie sich an Ihren Administrator, um Unterstützung zu erhalten.

Als Administrator können Sie das Gerät für einen anderen IAM-Benutzer deaktivieren. Dadurch kann der Benutzer sich ohne MFA anmelden. Dies kann eine vorübergehende Lösung darstellen, während die MFA-Gerät ersetzt wird oder wenn das Gerät vorübergehend nicht verfügbar ist. Wir empfehlen jedoch, dass Sie so bald wie möglich ein neues Gerät für den Benutzer aktivieren. Weitere Informationen zum Aktivieren eines neuen MFA-Geräts finden Sie unter the section called “Aktivieren von MFA-Geräten”.

> **Note**
>
> Wenn Sie die API verwenden oder einen Benutzer aus Ihrem löschen AWS CLI möchten AWS-Konto, müssen Sie das MFA-Gerät des Benutzers deaktivieren oder löschen. Diese Änderung nehmen Sie im Rahmen der Entfernung des Benutzers vor. Weitere Informationen zum Löschen von Benutzern finden Sie unter Verwalten von IAM-Benutzern.

Themen

Deaktivieren von MFA-Geräten (Konsole)

So deaktivieren Sie ein MFA-Gerät für einen anderen IAM-Benutzer (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 auf Users (Benutzer).
3. Zum Deaktivieren des MFA-Geräts für einen Benutzer wählen Sie den Namen des Benutzer aus, dessen MFA Sie entfernen möchten.
4. Wechseln Sie zur Registerkarte Sicherheitsanmeldeinformationen.
5. Wählen Sie unter Multi-factor authentication (MFA) (Multi-Faktor-Authentifizierung (MFA)) die Optionsschaltfläche neben dem MFA-Gerät, dann Remove (Entfernen)) und dann Remove (Entfernen).

   Das Gerät wurde entfernt von AWS. Es kann erst zur Anmeldung oder Authentifizierung von Anfragen verwendet werden, wenn es erneut aktiviert und einem AWS Benutzer zugeordnet wurde oder. Root-Benutzer des AWS-Kontos

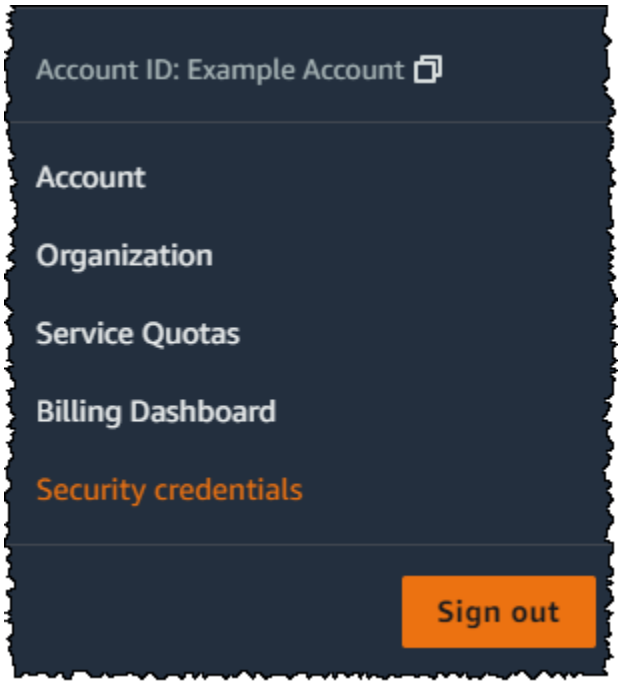
So deaktivieren Sie ein MFA-Gerät für Ihren Root-Benutzer des AWS-Kontos (Konsole)

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

   > **Note**
   >
   > Als Root-Benutzer können Sie sich nicht auf der Seite Sign in as IAM user (Als IAM-Benutzer anmelden) anmelden. Wenn Ihnen die Seite Sign in as IAM user (Als IAM-Benutzer anmelden) angezeigt wird, wählen Sie die Option Sign in using root user email (Mit Root-Benutzer-E-Mail anmelden) unten auf der Seite. Hilfe bei der Anmeldung als

Root-Benutzer finden Sie im Benutzerhandbuch unter AWS Management Console Als AWS-Anmeldung Root-Benutzer anmelden.

2. Klicken Sie rechts in der Navigationsleiste auf den Kontonamen und wählen Sie dann Security Credentials (Sicherheitsanmeldeinformationen). Sofern erforderlich, wählen Sie Continue to Security Credentials (Weiter zu Sicherheitsanmeldeinformationen).



3. Wählen Sie im Abschnitt Multi-Faktor-Authentifizierung (MFA) das Optionsfeld neben dem MFA-Gerät, das Sie deaktivieren möchten, und wählen Sie Remove (Entfernen).
4. Wählen Sie Remove (Entfernen) aus.

   Das MFA-Gerät wird für das AWS-Konto deaktiviert. Suchen Sie in der E-Mail, die mit Ihrer verknüpft ist, AWS-Konto nach einer Bestätigungsnachricht von Amazon Web Services. In der E-Mail-Nachricht werden Sie darüber informiert, dass Ihre Amazon Web Services Multi-Factor Authentication (MFA) deaktiviert wurde. Die Nachricht stammt von `@amazon.com` oder `@aws.amazon.com`.

Deaktivieren von MFA-Geräten (AWS CLI)

So deaktivieren Sie ein MFA-Gerät für einen IAM-Benutzer (AWS CLI)

- Führen Sie diesen Befehl aus: `aws iam deactivate-mfa-device`

Deaktivierung von MFA-Geräten (API)AWS

So deaktivieren Sie ein MFA-Gerät für einen IAM-Benutzer (API)AWS

- Rufen Sie diese Operation auf: `DeactivateMFADevice`

Was passiert, wenn ein MFA-Gerät verloren geht oder nicht funktioniert?

Wenn Ihr virtuelles MFA-Gerät oder Ihr Hardware-TOTP-Token ordnungsgemäß zu funktionieren scheint, Sie es aber nicht für den Zugriff auf Ihre AWS Ressourcen verwenden können, ist es möglicherweise nicht mehr synchronisiert mit. AWS Weitere Informationen zum Synchronisieren eines virtuellen MFA-Geräts oder Hardware-MFA-Geräts finden Sie unter Resynchronisieren von virtuellen und physischen MFA-Geräten. FIDO-Sicherheitsschlüssel verlieren ihre Synchronisation nicht.

Wenn Ihr Gerät Root-Benutzer des AWS-Kontos mit Multi-Faktor-Authentifizierung (MFA) verloren geht, beschädigt ist oder nicht funktioniert, können Sie den Zugriff auf Ihr Konto wiederherstellen. IAM-Benutzer müssen sich zur Deaktivierung des Geräts an einen Administrator wenden.

> **Important**
>
> Wir empfehlen Ihnen, mehrere MFA-Geräte für Ihre IAM-Benutzer zu aktivieren, damit Sie im Falle eines verlorenen oder unzugänglichen MFA-Geräts weiterhin auf Ihr Konto zugreifen können. Sie können bis zu acht MFA-Geräte einer beliebigen Kombination der derzeit unterstützten MFA-Typen für Ihren AWS-Konto -Root-Benutzer und Ihre IAM-Benutzer registrieren.

Wiederherstellen eines Stammbenutzer-MFA-Geräts

Wenn Ihr Root-Benutzer des AWS-Kontos Multi-Faktor-Authentifizierungsgerät (MFA) verloren geht, beschädigt ist oder nicht funktioniert, können Sie sich mit einem anderen MFA-Gerät anmelden, das für dasselbe registriert ist. Root-Benutzer des AWS-Kontos Wenn für den Root-Benutzer nur ein MFA-Gerät aktiviert ist, können Sie alternative Authentifizierungsmethoden verwenden. Das heißt, wenn Sie sich nicht mit Ihrem MFA-Gerät anmelden können, können Sie sich anmelden, indem Sie Ihre Identität mit der E-Mail und der Telefonnummer des Hauptansprechpartners, die in Ihrem Konto registriert sind, verifizieren.

Bevor Sie alternative Authentifizierungsfaktoren verwenden, um sich als Root-Benutzer anzumelden, müssen Sie auf die E-Mail und die primäre Telefonnummer zugreifen können, die Ihrem Konto zugeordnet sind. Wenn Sie die primäre Kontakttelefonnummer aktualisieren müssen, können Sie sich als IAM-Benutzer mit Administrator-Zugriff statt des Root-Benutzers anmelden. Weitere Anweisungen zur Aktualisierung der Kontokontaktinformationen finden Sie im AWS Billing -Benutzerhandbuch unter Kontaktinformationen bearbeiten. Wenn Sie keinen Zugriff auf eine E-Mail und eine primäre Kontakttelefonnummer haben, müssen Sie sich an den AWS Support wenden.

> **Important**
>
> Wir empfehlen Ihnen, die mit Ihrem Root-Benutzer verknüpfte E-Mail-Adresse und Kontakttelefonnummer auf dem neuesten Stand zu halten, damit Ihr Konto erfolgreich wiederhergestellt werden kann. Weitere Informationen finden Sie im AWS Account Management Referenzhandbuch unter Aktualisieren Sie den Hauptkontakt für Ihr AWS-Konto.

Um sich mit alternativen Authentifizierungsfaktoren anzumelden als 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.
2. Wählen Sie auf der Seite Zusätzliche Verifizierung erforderlich eine MFA-Methode für die Authentifizierung aus und klicken Sie auf Weiter.

   > **Note**
   >
   > Möglicherweise sehen Sie alternativen Text wie Sign in using MFA (Mit MFA anmelden), Troubleshoot your authentication device (Fehlerbehebung für Ihr Authentifizierungsgerät), oder Troubleshoot MFA (Fehlerbehebung für MFA), aber die Funktionalität ist dieselbe. Wenn Sie keine alternativen Authentifizierungsfaktoren verwenden können, um Ihre E-Mail-Adresse und die Telefonnummer Ihres Hauptansprechpartners zu verifizieren, wenden Sie sich an AWS Support, um Ihr MFA-Gerät zu deaktivieren.

3. Je nachdem, welche Art von MFA Sie verwenden, wird eine andere Seite angezeigt, aber die Option Fehlerbehebung bei MFA funktioniert genauso. Wählen Sie auf der Seite Zusätzliche

Überprüfung erforderlich oder Multifaktor-Authentifizierung die Option Fehlerbehebung bei MFA aus.

4. Geben Sie, falls erforderlich, das Passwort erneut ein, und wählen Sie Sign in (Anmelden).
5. Wählen Sie auf der Seite Fehlerbehebung bei Ihrem Authentifizierungsgerät im Abschnitt Anmelden mit alternativen Faktoren der Authentifizierung die Option Anmelden mit alternativen Faktoren.
6. Authentifizieren Sie Ihr Konto auf der Seite Anmeldung mit alternativen Faktoren, indem Sie die E-Mail-Adresse verifizieren, und wählen Sie Verifizierungs-E-Mail senden aus.
7. Suchen Sie in der E-Mail, die mit Ihrer verknüpft ist, AWS-Konto nach einer Nachricht von Amazon Web Services (recover-mfa-no-reply@verify .signin.aws). Befolgen Sie die Anweisungen in der E-Mail-Nachricht.

   Wenn in Ihrem Konto keine E-Mail-Nachricht angezeigt wird, überprüfen Sie Ihren Spam-Ordner oder wählen Sie in Ihrem Browser Resend the email (E-Mail erneut versenden).
8. Nachdem Ihre E-Mail-Adresse verifiziert wurde, können Sie mit der Authentifizierung Ihres Kontos fortfahren. Zum Überprüfen Ihrer Haupttelefonnummer wählen Sie Call me now (Mich jetzt anrufen).
9. Nehmen Sie den Anruf von entgegen AWS und geben Sie, wenn Sie dazu aufgefordert werden, die sechsstellige Nummer von der AWS Website auf der Telefontastatur ein.

   Wenn Sie keinen Anruf von erhalten AWS, wählen Sie Anmelden, um sich erneut an der Konsole anzumelden und von vorne zu beginnen. Oder sehen Sie unter Verlorenes oder unbrauchbares Multi-Faktor-Authentifizierung (MFA)-Gerät nach, um den Support für Hilfe zu kontaktieren.
10. Nach dem Verifizieren Ihrer Telefonnummer können Sie sich bei Ihrem Konto anmelden, indem Sie Sign in to the console (Anmelden an der Konsole) auswählen.
11. Der nächste Schritt variiert je nach Art der MFA, die Sie verwenden:

    - Für ein virtuelles MFA-Gerät, entfernen Sie das Konto von Ihrem Gerät. Löschen Sie dann auf der Seite AWS -Sicherheitsanmeldeinformationen die Entität für das alte virtuelle MFA-Gerät, bevor Sie eine neue erstellen.
    - Für einen FIDO-Sicherheitsschlüssel gehen Sie auf die Seite AWS - Sicherheitsanmeldeinformationen und deaktivieren Sie den alten FIDO-Sicherheitsschlüssel, bevor Sie einen neuen aktivieren.
    - Für ein physisches TOTP-Token wenden Sie sich an den entsprechenden Anbieter, um das Gerät zu reparieren oder auszutauschen. Sie können sich weiter mit der

alternativen Authentifizierung anmelden, bis Sie Ihr neues Gerät erhalten. Nachdem Sie ein neues Hardware-MFA-Gerät erhalten haben, können Sie auf der Seite AWS - Sicherheitsanmeldeinformationen die Entität für das alte MFA-Gerät löschen, bevor Sie eine neue erstellen.

> **Note**
>
> Sie müssen ein verlorenes oder gestohlenes MFA-Gerät nicht durch ein gleichartiges Gerät ersetzen. Wenn beispielsweise Ihr FIDO-Sicherheitsschlüssel kaputt gegangen ist und Sie einen neuen bestellen, können Sie ein virtuelles MFA-Gerät oder ein physisches TOTP-Token verwenden, bis Sie einen neuen FIDO-Sicherheitsschlüssel erhalten.

> **Important**
>
> Wenn Ihr MFA-Gerät verloren gegangen ist oder gestohlen wurde, sollten Sie Ihr Root-Benutzerpasswort ändern, nachdem Sie sich mit alternativen Authentifizierungsfaktoren angemeldet und Ihr MFA-Ersatzgerät eingerichtet haben, um sich dagegen abzusichern, dass ein Angreifer das Authentifizierungsgerät gestohlen hat und möglicherweise auch Ihr aktuelles Passwort kennt. Weitere Informationen finden Sie im Referenzhandbuch zu AWS Account Management unter Change the password for the Root-Benutzer des AWS-Kontos.

Wiederherstellen eines IAM-Benutzer-MFA-Geräts

Wenn Sie ein IAM-Benutzer sind und Ihr Gerät verloren geht oder nicht mehr funktioniert, können Sie es nicht selbst wiederherstellen. Sie müssen sich zur Deaktivierung des Geräts an einen Administrator wenden. Anschließend können Sie ein neues Gerät aktivieren.

So erhalten Sie als IAM-Benutzer Hilfe für ein MFA-Gerät

1. Wenden Sie sich an den AWS Administrator oder eine andere Person, die Ihnen den Benutzernamen und das Passwort für den IAM-Benutzer gegeben hat. Der Administrator muss das MFA-Gerät deaktivieren, wie in Deaktivieren von MFA-Geräten beschrieben, damit Sie sich anmelden können.
2. Der nächste Schritt variiert je nach Art der MFA, die Sie verwenden:

- Für ein virtuelles MFA-Gerät, entfernen Sie das Konto von Ihrem Gerät. Aktivieren Sie dann das virtuelle Gerät wie in Aktivieren eines virtuellen Multi-Factor Authentication (MFA)-Geräts (Konsole) beschrieben.
- Bei einem FIDO-Sicherheitsschlüssel, wenden Sie sich an den entsprechenden Anbieter, um das Gerät zu reparieren oder auszutauschen. Wenn Sie den neuen FIDO-Sicherheitsschlüssel erhalten, aktivieren Sie ihn wie in Aktivieren eines Hauptschlüssels oder Sicherheitsschlüssels (Konsole) beschrieben.
- Für ein physisches TOTP-Token wenden Sie sich an den entsprechenden Anbieter, um das Gerät zu reparieren oder auszutauschen. Nachdem Sie das neue physische MFA-Gerät erhalten haben, aktivieren Sie das Gerät wie in Aktivieren eines Hardware-TOTP-Tokens (Konsole) beschrieben.

> **Note**
>
> Sie müssen ein verlorenes oder gestohlenes MFA-Gerät nicht durch ein gleichartiges Gerät ersetzen. Sie können bis zu acht MFA-Geräte beliebiger Kombination verwenden. Wenn beispielsweise Ihr FIDO-Sicherheitsschlüssel kaputt gegangen ist und Sie einen neuen bestellen, können Sie ein virtuelles MFA-Gerät oder ein physisches TOTP-Token verwenden, bis Sie einen neuen FIDO-Sicherheitsschlüssel erhalten.

3. Wenn Ihr MFA-Gerät verloren gegangen ist oder gestohlen wurde, ändern Sie auch Ihr Passwort, falls ein Angreifer das Authentifizierungsgerät gestohlen hat und möglicherweise auch Ihr aktuelles Passwort kennt. Weitere Informationen finden Sie unter Verwalten von Passwörtern für IAM-Benutzer

Konfigurieren eines MFA-geschützten API-Zugriffs

Mit IAM-Richtlinien können Sie angeben, welche API-Operationen ein Benutzer aufrufen darf. In einigen Fällen möchten Sie ggf. die zusätzliche Sicherheit in Anspruch nehmen, die Ihnen die AWS Multi-Factor Authentication (MFA) als Voraussetzung bietet, damit der Benutzer besonders sensible Aktionen durchführen darf.

Sie verfügen beispielsweise über eine Richtlinie, die dem Benutzer gestattet, die Amazon EC2 Aktionen `RunInstances`, `DescribeInstances` und `StopInstances` auszuführen. Möglicherweise möchten Sie jedoch eine destruktive Aktion einschränken `TerminateInstances`

und sicherstellen, dass Benutzer diese Aktion nur ausführen können, wenn sie sich mit einem AWS MFA-Gerät authentifizieren.

Themen

Übersicht

Um den MFA-Schutz zu API-Operationen hinzuzufügen, sind folgende Schritte auszuführen:

1. Der Administrator konfiguriert ein AWS MFA-Gerät für jeden Benutzer, der API-Anfragen stellen muss, für die eine MFA-Authentifizierung erforderlich ist. Dieser Prozess wird in Aktivierung von MFA-Geräten für Benutzer in AWS beschrieben.
2. Der Administrator erstellt Richtlinien für die Benutzer, die ein `Condition` Element enthalten, das überprüft, ob sich der Benutzer mit einem AWS MFA-Gerät authentifiziert hat.
3. Der Benutzer ruft eine der AWS STS API-Operationen auf, die die MFA-Parameter unterstützen GetSessionToken, AssumeRoleoder, je nach Szenario für den MFA-Schutz, wie später erklärt. Als Teil des Aufrufs schließt der Benutzer die mit dem Benutzer verknüpfte Gerät-ID für das Gerät ein. Der Benutzer schließt außerdem das zeitbasierte Einmalpasswort (Time-based One-Time Password, TOTP) ein. In beiden Fällen erhält der Benutzer temporäre Sicherheitsanmeldeinformationen zurück, die der Benutzer für zusätzliche Anfragen an AWS benutzen kann.

> **Note**
>
> MFA-Schutz für die API-Operationen eines Service ist nur dann verfügbar, wenn der Service temporäre Sicherheitsanmeldeinformationen unterstützt. Eine Liste dieser Services finden Sie unter Verwendung temporärer Sicherheitscodes für den Zugriff auf AWS.

Wenn die Autorisierung fehlschlägt, wird die Fehlermeldung „Zugriff verweigert“ AWS zurückgegeben (wie bei jedem nicht autorisierten Zugriff). Wenn MFA-geschützte API-Richtlinien vorhanden sind, wird der Zugriff auf die in den Richtlinien angegebenen AWS API-Operationen verweigert, wenn der

Benutzer versucht, eine API-Operation ohne gültige MFA-Authentifizierung aufzurufen. Die Operation wird auch verweigert, wenn der Zeitstempel der Anforderung für die API-Operation sich außerhalb des in der Richtlinie festgelegten zulässigen Bereichs befindet. Der Benutzer muss sich erneut mit MFA authentifizieren, indem neue temporäre Sicherheitsanmeldeinformationen mit einem MFA-Code und einer Geräte-Seriennummer angefordert werden.

IAM-Richtlinien mit MFA-Bedingungen

Richtlinien mit MFA-Bedingungen können folgenden Elementen angefügt werden:

- Einem IAM-Benutzer oder einer IAM-Gruppe
- Einer Ressource, wie zum Beispiel ein Amazon S3-Bucket, eine Amazon SQS-Warteschlange oder ein Amazon SNS-Thema
- Die Vertrauensrichtlinie einer IAM-Rolle, die von einem Benutzer übernommen werden kann

Sie können eine MFA-Bedingung in einer Richtlinie zum Überprüfen der folgenden Eigenschaften verwenden:

- Vorhanden – Um mit MFA lediglich zu überprüfen, ob sich der Benutzer mit MFA authentifiziert hat, prüfen Sie, dass der `aws:MultiFactorAuthPresent`-Schlüssel auf `True` gesetzt ist, und zwar einer `Bool`-Bedingung. Der Schlüssel ist nur vorhanden, wenn sich der Benutzer mit kurzfristigen Anmeldeinformationen authentifiziert. Langfristige Anmeldeinformationen wie Zugriffsschlüssel enthalten diesen Schlüssel nicht.
- Dauer – Wenn Sie den Zugriff nur innerhalb einer bestimmten Zeit nach der MFA-Authentifizierung gewähren möchten, verwenden Sie einen numerischen Bedingungstyp, um das Alter des `aws:MultiFactorAuthAge`-Schlüssels mit einem Wert (wie 3.600 Sekunden) zu vergleichen. Beachten Sie, dass der `aws:MultiFactorAuthAge`-Schlüssel nicht vorhanden ist, wenn keine MFA verwendet wurde.

Das folgende Beispiel zeigt die Vertrauensrichtlinie einer IAM-Rolle, die eine MFA-Bedingung enthält, um die Existenz der MFA-Authentifizierung festzustellen. Mit dieser Richtlinie können Benutzer der im `Principal` Element AWS-Konto angegebenen Rolle (durch eine gültige AWS-Konto ID `ACCOUNT-B-ID` ersetzen) die Rolle übernehmen, der diese Richtlinie zugewiesen ist. Solche Benutzer können die Rolle jedoch nur annehmen, wenn sie sich mit MFA authentifizieren.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Weitere Informationen über die Bedingungstypen für MFA finden Sie unter AWS Kontextschlüssel für globale Bedingungen, Numerische Bedingungsoperatoren und Bedingungsoperator zur Prüfung der Existenz von Bedingungsoperatoren .

Sie haben die Wahl zwischen GetSessionToken und AssumeRole

AWS STS bietet zwei API-Operationen, mit denen Benutzer MFA-Informationen weitergeben können: `GetSessionToken` und`AssumeRole`. Die vom Benutzer zur Erlangung temporärer Sicherheitsanmeldeinformationen aufgerufene API-Operation wird durch eines der folgenden Szenarien bestimmt.

Verwenden Sie **`GetSessionToken`** für die folgenden Szenarien:

- Rufen Sie API-Operationen auf, die auf Ressourcen zugreifen, genauso AWS-Konto wie der IAM-Benutzer, der die Anfrage stellt. Beachten Sie, dass temporäre Anmeldeinformationen aus einer `GetSessionToken` Anfrage nur dann auf IAM- und AWS STS API-Operationen zugreifen können, wenn Sie MFA-Informationen in die Anforderung von Anmeldeinformationen aufnehmen. Da von `GetSessionToken` zurückgegebene temporäre Anmeldeinformationen MFA-Informationen enthalten, können Sie mit den Anmeldeinformationen ausgeführte einzelne API-Operationen auf MFA prüfen.
- Zugriff auf Ressourcen, die über ressourcenbasierte Richtlinien geschützt sind, die eine MFA-Bedingung enthalten.

Der Zweck der Operation `GetSessionToken` besteht darin, den Benutzer mithilfe von MFA zu authentifizieren. Richtlinien können nicht dazu verwendet werden, Authentifizierungsoperationen zu steuern.

Verwenden Sie **`AssumeRole`** für die folgenden Szenarien:

- Aufrufe von API-Operationen, die auf die Ressourcen in demselben oder einem anderen AWS-Konto zugreifen. Die API-Aufrufe können jedes IAM oder jede API beinhalten. AWS STS Beachten Sie, dass MFA zu dem Zeitpunkt, zu dem der Benutzer die Rolle übernimmt, aktiviert

ist. Die von `AssumeRole` zurückgegebenen temporären Anmeldeinformationen enthalten keine MFA-Informationen im Kontext, sodass einzelne API-Operationen nicht auf MFA geprüft werden können. Daher müssen Sie `GetSessionToken` verwenden, um den Zugriff auf die von ressourcenbasierten Richtlinien geschützten Ressourcen einzuschränken.

Weitere Informationen zur Implementierung dieser Szenarien finden Sie weiter unten in diesem Dokument.

Wichtige Punkte für den MFA-geschützten API-Zugriff

Beachten Sie die folgenden Aspekte in Bezug auf den MFA-Schutz für API-Operationen:

- MFA-Schutz steht nur mit temporären Sicherheitsanmeldeinformationen zur Verfügung, die mit `AssumeRole` oder `GetSessionToken` eingeholt werden müssen.
- Sie können den MFA-geschützten API-Zugriff nicht mit Root-Benutzer des AWS-Kontos Anmeldeinformationen verwenden.
- Sie können den MFA-geschützten API-Zugriff nicht mit U2F-Sicherheitsschlüsseln verwenden.
- Verbundbenutzern kann kein MFA-Gerät zur Verwendung mit AWS Diensten zugewiesen werden, sodass sie nicht auf AWS Ressourcen zugreifen können, die von MFA gesteuert werden. (Siehe nächster Punkt.)
- Andere AWS STS API-Operationen, die temporäre Anmeldeinformationen zurückgeben, unterstützen MFA nicht. Für `AssumeRoleWithWebIdentity` und `AssumeRoleWithSAML` wird der Benutzer von einem externen Anbieter authentifiziert und AWS kann nicht feststellen, ob dieser Anbieter MFA benötigt. Für `GetFederationToken` wird MFA nicht notwendigerweise mit einem spezifischen Benutzer verknüpft.
- Langfristige Anmeldeinformationen (IAM-Benutzer-Zugriffsschlüssel und Stammbenutzer-Zugriffsschlüssel) können für den MFA-geschützten API-Zugriff ebenfalls nicht verwendet werden, da sie zeitlich unbegrenzt sind.
- `AssumeRole` und `GetSessionToken` können auch ohne MFA-Informationen aufgerufen werden. In diesem Fall erhält der Aufrufer temporäre Sicherheitsanmeldeinformationen, aber die Sitzungsinformationen dieser Anmeldeinformationen enthalten keine Angaben darüber, ob der Benutzer sich mit MFA authentifiziert hat.
- Um MFA-Schutz für API-Operationen einzurichten, fügen Sie den Richtlinien MFA-Bedingungen hinzu. In einer Richtlinie muss der `aws:MultiFactorAuthPresent`-Bedingungsschlüssel enthalten sein, damit die Verwendung von MFA durchgesetzt wird. Bei der kontoübergreifenden Delegierung muss die Vertrauensrichtlinie der Rolle den Bedingungsschlüssel enthalten.

- Wenn Sie einer anderen Person AWS-Konto den Zugriff auf Ressourcen in Ihrem Konto gestatten, hängt die Sicherheit Ihrer Ressourcen von der Konfiguration des vertrauenswürdigen Kontos ab (das andere Konto, nicht Ihres). Dies gilt auch, wenn Sie eine Multi-Factor Authentication verlangen. Jede Identität im vertrauenswürdigen Konto mit Berechtigung zum Erstellen von virtuellen MFA-Geräten kann einen MFA-Anspruch erstellen, um den entsprechenden Teil der Vertrauensrichtlinie Ihrer Rolle zu erfüllen. Bevor Sie Mitgliedern eines anderen Kontos Zugriff auf Ihre AWS Ressourcen gewähren, für die eine Multi-Faktor-Authentifizierung erforderlich ist, sollten Sie sicherstellen, dass der Besitzer des vertrauenswürdigen Kontos die bewährten Sicherheitsmethoden befolgt. Beispiel: Das vertrauenswürdige Konto sollte den Zugriff auf sensible API-Operationen, z. B. API-Operationen zur MFA-Geräteverwaltung, auf spezifische, vertrauenswürdige Identitäten beschränken.
- Wenn eine Richtlinie eine MFA-Bedingung enthält, wird eine Anfrage abgelehnt, falls die Benutzer nicht mit MFA authentifiziert worden sind oder die MFA-Geräte-ID oder das zeitlich begrenzte, einmalige Passwort ungültig ist.

Szenario: MFA-Schutz für kontoübergreifende Delegierung

In diesem Szenario möchten Sie den Zugriff an IAM-Benutzer in einem anderen Konto delegieren, aber nur, wenn die Benutzer mit einem AWS MFA-Gerät authentifiziert sind. Weitere Informationen über die kontoübergreifende Delegierung finden Sie unter Rollenbegriffe und -konzepte.

Angenommen, Sie haben ein Konto A (das vertrauende Konto, das die zu gewünschten Ressourcen enthält) mit der IAM-Benutzerin Anaya, die über Administratorberechtigungen verfügt. Sie möchten dem Benutzer Richard im Konto B (dem vertrauenswürdigen Konto) Zugriff gewähren, dabei aber sicherstellen, dass sich Richard mit MFA authentifiziert, bevor er die Rolle übernimmt.

1. Im vertrauenswürdigen Konto A erstellt Anaya eine IAM-Rolle mit dem Namen `CrossAccountRole` und legt den Principal in der Vertrauensrichtlinie der Rolle auf die Konto-ID von Konto B fest. Die Vertrauensrichtlinie erteilt die Genehmigung für die Aktion. AWS STS `AssumeRole` Darüber hinaus fügt Anaya der Vertrauensrichtlinie eine MFA-Bedingung hinzu, wie im folgenden Beispiel dargestellt.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
```

```
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

2. Anaya fügt der Rolle eine Berechtigungsrichtlinie hinzu, die die Ausführungsberechtigungen enthält. Die Berechtigungsrichtlinie für eine Rolle mit MFA-Schutz unterscheidet sich nicht von anderen Berechtigungsrichtlinien für Rollen. Das folgende Beispiel zeigt die Richtlinie, die Anaya zur Rolle hinzufügt. Sie gestattet einem Benutzer, der diese Richtlinie übernimmt, beliebige Amazon-DynamoDB-Aktionen in der Tabelle `Books` in Konto A durchzuführen. Diese Richtlinie gestattet auch die `dynamodb:ListTables`-Aktion, die zum Ausführen der Aktionen in der Konsole erforderlich ist.

> **Note**
>
> Die Berechtigungsrichtlinie enthält keine MFA-Bedingung. Beachten Sie dabei, dass die MFA-Authentifizierung nur verwendet wird, um zu ermitteln, ob ein Benutzer die Rolle übernehmen kann. Sobald der Benutzer die Rolle übernommen hat, werden keine weiteren MFA-Kontrollen durchgeführt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. Im vertrauenswürdigen Konto B stellt der Administrator sicher, dass der IAM-Benutzer Richard mit einem AWS MFA-Gerät konfiguriert ist und dass er die ID des Geräts kennt. Die Gerät-ID ist bei

einem physischen MFA-Gerät die Seriennummer oder bei einem virtuellen MFA-Gerät der Geräte-ARN.

4. In Konto B fügt der Administrator die folgenden Richtlinien dem Benutzer Richard an (oder einer Gruppe, deren Mitglied Bob ist), die ihm das Aufrufen der Aktion `AssumeRole` gestatten. Die Ressource ist auf den ARN der von Anaya in Schritt 1 erstellten Rolle festgelegt. Beachten Sie, dass diese Richtlinie keine MFA-Bedingung enthält.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sts:AssumeRole"],
    "Resource": ["arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole"]
  }]
}
```

5. Richard (oder eine von Richard ausgeführte Anwendung) ruft im Konto B `AssumeRole` auf. Der API-Aufruf enthält den ARN der zu übernehmenden Rolle (`arn:aws:iam::`*`ACCOUNT-A-ID`*`:role/CrossAccountRole`), die MFA-Geräte-ID und das aktuelle TOTP, das Richard von seinem Gerät erhält.

   Wenn Richard anruft`AssumeRole`, AWS stellt er fest, ob er über gültige Anmeldeinformationen verfügt, einschließlich der Anforderungen für MFA. Wenn dies der Fall ist, übernimmt Richard die Rolle und kann beliebige DynamoDB-Aktionen in der Tabelle mit dem Namen `Books` im Konto A ausführen, sofern er die temporären Anmeldeinformationen der Rolle verwendet.

   Ein Beispiel für ein Programm, das `AssumeRole` aufruft, finden Sie unter Telefonieren AssumeRole mit MFA-Authentifizierung.

Szenario: MFA-Schutz für Zugriff auf API-Operationen im aktuellen Konto

In diesem Szenario sollten Sie sicherstellen, dass ein Benutzer in Ihrem Konto nur dann auf sensible API-Operationen zugreifen AWS-Konto kann, wenn der Benutzer mit einem AWS MFA-Gerät authentifiziert ist.

Angenommen, Sie haben ein Konto A mit einer Gruppe von Entwicklern, die mit EC2-Instances arbeiten müssen. Normale Entwickler können mit den Instances arbeiten, aber sie haben keine Berechtigungen für die Aktion `ec2:StopInstances` oder `ec2:TerminateInstances`. Sie möchten diese besonderen, "destruktiven" Aktionen auf nur einige wenige vertrauenswürdige

Benutzer beschränken. Daher fügen Sie einen MFA-Schutz der Richtlinie hinzu, der diese sensiblen Amazon EC2-Aktionen zulässt.

In diesem Szenario ist Sofía einer dieser vertrauenswürdigen Benutzer. Die Benutzerin Anaya ist eine Administratorin in Konto A.

1. Anaya stellt sicher, dass Sofía mit einem AWS MFA-Gerät konfiguriert ist und dass Sofía die ID des Geräts kennt. Die Gerät-ID ist bei einem physischen MFA-Gerät die Seriennummer oder bei einem virtuellen MFA-Gerät der Geräte-ARN.
2. Anaya erstellt eine Gruppe mit dem Namen `EC2-Admins` und fügt Sofía dieser Gruppe hinzu.
3. Anaya fügt die folgende Richtlinie an die Gruppe `EC2-Admins` an. Diese Richtlinie gewährt Benutzern die Berechtigung zum Aufrufen der Amazon EC2-Aktionen `StopInstances` und `TerminateInstances` nur dann, wenn sich der Benutzer mit MFA authentifiziert hat.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
    ],
    "Resource": ["*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.
> **Note**
>
> Damit diese Richtlinie wirksam wird, müssen sich Benutzer zuerst abmelden und dann wieder anmelden.

   Wenn Benutzer Sofía eine Amazon EC2-Instance anhalten oder beenden muss, ruft sie (oder eine von ihr ausgeführte Anwendung) `GetSessionToken` auf. Diese API-Operation übergibt die ID des MFA-Geräts und das aktuelle TOTP, das Sofía von ihrem Gerät erhält.
5. Die Benutzerin Sofía (oder eine von Sofía verwendete Anwendung) verwendet die von `GetSessionToken` bereitgestellten Anmeldeinformationen, um die Amazon EC2-Aktion `StopInstances` oder `TerminateInstances` aufzurufen.

Ein Beispiel für ein Programm, das `GetSessionToken` aufruft, finden Sie unter Telefonieren GetSessionToken mit MFA-Authentifizierung weiter unten in diesem Dokument.

Szenario: MFA-Schutz für Ressourcen mit ressourcenbasierten Richtlinien

In diesem Szenario sind Sie der Besitzer eines S3-Buckets, einer SQS-Warteschlange oder eines SNS-Themas. Sie möchten sicherstellen, dass jeder Benutzer, der auf AWS-Konto die Ressource zugreift, von einem AWS MFA-Gerät authentifiziert wird.

Dieses Szenario zeigt eine Möglichkeit, den kontoübergreifenden MFA-Schutz herzustellen, ohne dass die Benutzer eine Rolle übernehmen müssen. In diesem Fall kann der Benutzer auf die Ressource zugreifen, wenn drei Bedingungen erfüllt sind: Der Benutzer muss sich mithilfe von MFA authentifizieren, muss in der Lage sein, temporäre Sicherheitsanmeldeinformationen über `GetSessionToken` abzurufen, und muss über ein Konto verfügen, das von der Ressourcenrichtlinie als vertrauenswürdig eingestuft wird.

Angenommen, Sie befinden sich im Konto A und erstellen einen S3-Bucket. Sie möchten Benutzern, die sich in mehreren verschiedenen Ländern befinden, Zugriff auf diesen Bucket gewähren AWS-Konten, aber nur, wenn diese Benutzer mit MFA authentifiziert sind.

In diesem Szenario ist Anaya eine Administratorin im Konto A und Benutzer Nikhil ist ein IAM-Benutzer im Konto C.

1. Anaya erstellt im Konto A einen Bucket mit dem Namen `Account-A-bucket`.
2. Anaya fügt die Bucket-Richtlinie dem Bucket hinzu. Die Richtlinie gestattet jedem Benutzer in Konto A, Konto B und Konto C, die Amazon S3-Aktionen `PutObject` und `DeleteObject` im S3-Bucket auszuführen. Die Richtlinie enthält eine MFA-Bedingung.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
```

```
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3; bietet eine MFA Delete-Funktion (nur) für den Stamm-Kontozugriff. Sie können Amazon S3 MFA Delete aktivieren, wenn Sie den Versionierungsstatus des Buckets festlegen. Amazon S3 MFA Delete kann nicht auf einen IAM-Benutzer angewendet werden und wird unabhängig von MFA-geschützten API-Zugriffen verwaltet. Ein IAM-Benutzer mit Berechtigungen zum Löschen eines Buckets kann keinen Bucket löschen, für den Amazon S3 MFA Delete aktiviert ist. Weitere Informationen zu Amazon S3 MFA Delete finden Sie unter MFA Delete.

3. Im Konto C stellt ein Administrator sicher, dass für den Benutzer Nikhil ein AWS -MFA-Gerät konfiguriert ist und er die Geräte-ID kennt. Die Gerät-ID ist bei einem physischen MFA-Gerät die Seriennummer oder bei einem virtuellen MFA-Gerät der Geräte-ARN.
4. Nikhil (oder eine von Nikhil ausgeführte Anwendung) ruft im Konto C `GetSessionToken` auf. Der Aufruf enthält die ID oder den ARN des MFA-Geräts und das aktuelle TOTP, das Nikhil von seinem Gerät erhält.
5. Nikhil (oder eine von Nikhil verwendete Anwendung) benutzt die von `GetSessionToken` bereitgestellten Anmeldeinformationen, um die Amazon S3–`PutObject`Aktion zum Hochladen einer Datei nach `Account-A-bucket` aufzurufen.

   Ein Beispiel für ein Programm, das `GetSessionToken` aufruft, finden Sie unter Telefonieren GetSessionToken mit MFA-Authentifizierung weiter unten in diesem Dokument.

> **Note**
>
> Die von `AssumeRole` zurückgegebenen temporären Anmeldeinformationen funktionieren in diesem Fall nicht. Obwohl der Benutzer MFA-Informationen bereitstellen kann, um eine Rolle zu übernehmen, enthalten die von `AssumeRole` zurückgegebenen temporären Anmeldeinformationen nicht die MFA-Informationen. Diese Informationen sind zur Erfüllung der MFA-Bedingung in der Richtlinie erforderlich.

Beispielcode: Anfordern von Anmeldeinformationen mit Multi-Factor Authentication

Die folgenden Beispiele veranschaulichen, wie die Operationen `GetSessionToken` und `AssumeRole` aufgerufen und MFA-Authentifizierungsparameter übergeben werden. Es werden keine Berechtigungen zum Aufrufen von `GetSessionToken` benötigt, Sie müssen jedoch über eine Richtlinie verfügen, die Ihnen gestattet, `AssumeRole` aufzurufen. Die zurückgegebenen Anmeldeinformationen werden dann verwendet, um alle S3-Buckets im Konto aufzulisten.

Telefonieren GetSessionToken mit MFA-Authentifizierung

Das folgende Beispiel zeigt, wie Sie `GetSessionToken` aufrufen und MFA-Authentifizierungsdaten übergeben. Die von der Operation `GetSessionToken` zurückgegebenen temporären Sicherheitsanmeldeinformationen werden dann verwendet, um alle S3-Buckets im Konto aufzulisten.

Die Richtlinie, die dem den Code ausführenden Benutzer (oder einer Gruppe, der er angehört) zugeordnet ist, stellt die Berechtigungen für die zurückgegeben temporären Anmeldeinformationen bereit. Für diesen Beispielcode muss die Richtlinie dem Benutzer die Erlaubnis erteilen, den Vorgang Amazon S3 `ListBuckets` anzufordern.

Die folgenden Codebeispiele zeigen die Verwendung`GetSessionToken`.

CLI

AWS CLI

So erhalten Sie einen Satz kurzfristiger Anmeldeinformationen für eine IAM-Identität

Der folgende `get-session-token`-Befehl ruft einen Satz kurzfristiger Anmeldeinformationen für die IAM-Identität ab, die den Aufruf ausführt. Die resultierenden Anmeldeinformationen können für Anfragen verwendet werden, bei denen die Richtlinie eine Multi-Faktor-Authentifizierung (MFA) erfordert. Die Anmeldeinformationen verfallen 15 Minuten nach ihrer Generierung.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Ausgabe:

```
{
```

```
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Weitere Informationen finden Sie unter Anfordern von temporären Sicherheitsanmeldeinformationen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetSessionTokenin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Gibt eine **`Amazon.RuntimeAWSCredentials`** Instanz zurück, die temporäre Anmeldeinformationen enthält, die für einen bestimmten Zeitraum gültig sind. Die Anmeldeinformationen, die zum Anfordern temporärer Anmeldeinformationen verwendet werden, werden aus den aktuellen Shell-Standardeinstellungen abgeleitet. Um andere Anmeldeinformationen anzugeben, verwenden Sie die Parameter - ProfileName oder - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Ausgabe:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Beispiel 2: Gibt eine **`Amazon.RuntimeAWSCredentials`** Instanz zurück, die temporäre Anmeldeinformationen enthält, die für eine Stunde gültig sind. Die für die Anfrage verwendeten Anmeldeinformationen stammen aus dem angegebenen Profil.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Ausgabe:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Beispiel 3: Gibt eine **`Amazon.RuntimeAWSCredentials`** Instanz mit temporären Anmeldeinformationen zurück, die für eine Stunde gültig sind. Dabei werden die Identifikationsnummer des MFA-Geräts, das dem Konto zugeordnet ist, dessen Anmeldeinformationen im Profil „myprofilename“ angegeben sind, und der vom Gerät bereitgestellte Wert verwendet.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Ausgabe:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Einzelheiten zur API finden Sie unter GetSessionTokenCmdlet-Referenz.AWS Tools for PowerShell

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie ein Sitzungs-Token ab, indem Sie ein MFA-Token übergeben, und verwenden Sie es, um Amazon-S3-Buckets für das Konto aufzulisten.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
```

```
    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Einzelheiten zur API finden Sie GetSessionTokenin AWS SDK for Python (Boto3) API Reference.

Telefonieren AssumeRole mit MFA-Authentifizierung

Die folgenden Beispiele zeigen, wie Sie `AssumeRole` aufrufen und MFA-Authentifizierungsdaten übergeben. Die von `AssumeRole` zurückgegebenen temporären Sicherheitsanmeldeinformationen werden dann verwendet, um alle Amazon S3-Buckets im Konto aufzulisten.

Weitere Informationen zu diesem Szenario finden Sie unter Szenario: MFA-Schutz für kontoübergreifende Delegierung.

Die folgenden Codebeispiele zeigen die Verwendung`AssumeRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
```

```
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);
```

```
            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
```

```
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
#############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
```

```
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
```

```
                              const Aws::String &externalId,
                              Aws::Auth::AWSCredentials &credentials,
                              const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So übernehmen Sie eine Rolle

Der folgende `assume-role`-Befehl ruft eine Reihe von kurzfristigen Anmeldeinformationen für die IAM-Rolle `s3-access-example` ab.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Ausgabe:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

Die Ausgabe des Befehls enthält einen Zugriffsschlüssel, einen geheimen Schlüssel und ein Sitzungs-Token, die Sie zur Authentifizierung bei AWS verwenden können.

Für die Verwendung AWS über die CLI können Sie ein benanntes Profil einrichten, das einer Rolle zugeordnet ist. Wenn Sie das Profil verwenden, ruft die AWS CLI assume-role auf und verwaltet die Anmeldeinformationen für Sie. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle in der AWS CLI im AWS CLI-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AssumeRole AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
```

```
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
```

```
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie den Client.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Übernehmen Sie die IAM-Rolle.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);
```

```
  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Gibt einen Satz temporärer Anmeldeinformationen (Zugriffsschlüssel, geheimer Schlüssel und Sitzungstoken) zurück, die eine Stunde lang für den Zugriff auf AWS Ressourcen verwendet werden können, auf die der anfragende Benutzer normalerweise keinen Zugriff hat. Die zurückgegebenen Anmeldeinformationen verfügen über die Berechtigungen, die durch die Zugriffsrichtlinie der übernommenen Rolle und die angegebene Richtlinie zulässig sind (Sie können die bereitgestellte Richtlinie nicht verwenden, um Berechtigungen zu gewähren, die über die in der Zugriffsrichtlinie der übernommenen Rolle definierten Berechtigungen hinausgehen).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Beispiel 2: Gibt einen Satz temporärer Anmeldeinformationen zurück, die für eine Stunde gültig sind und dieselben Berechtigungen haben, die in der Zugriffsrichtlinie der Rolle definiert sind, die angenommen wird.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Beispiel 3: Gibt einen Satz temporärer Anmeldeinformationen zurück, die die Seriennummer und das generierte Token aus einem MFA enthalten, das den Benutzeranmeldedaten zugeordnet ist, die zur Ausführung des Cmdlets verwendet wurden.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Beispiel 4: Gibt einen Satz temporärer Anmeldeinformationen zurück, die eine in einem Kundenkonto definierte Rolle übernommen haben. Für jede Rolle, die der Drittanbieter übernehmen kann, muss das Kundenkonto eine Rolle mithilfe einer Kennung erstellen, die bei jeder Übernahme der Rolle im ExternalId Parameter - übergeben werden muss.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Einzelheiten zur API finden Sie unter AssumeRole AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Übernehmen Sie eine IAM-Rolle, die ein MFA-Token erfordert, und verwenden Sie temporäre Anmeldeinformationen, um Amazon-S3-Buckets für das Konto aufzulisten.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
```

```
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Einzelheiten zur API finden Sie AssumeRolein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
                async throws -> STSClientTypes.Credentials {
    let input = AssumeRoleInput(
        roleArn: role.arn,
        roleSessionName: sessionName
    )
    do {
        let output = try await stsClient.assumeRole(input: input)

        guard let credentials = output.credentials else {
            throw ServiceHandlerError.authError
        }

        return credentials
    } catch {
        throw error
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der API-Referenz zum AWS SDK für Swift.

Finden Sie ungenutzte AWS Zugangsdaten

Um Ihre Sicherheit zu erhöhen AWS-Konto, entfernen Sie nicht benötigte IAM-Benutzeranmeldedaten (d. h. Passwörter und Zugriffsschlüssel). Wenn Benutzer beispielsweise Ihr Unternehmen verlassen oder keinen AWS Zugriff mehr benötigen, suchen Sie nach den Anmeldeinformationen, die sie verwendet haben, und stellen Sie sicher, dass sie nicht mehr betriebsbereit sind. Im Idealfall löschen Sie die Anmeldeinformationen, wenn sie nicht mehr benötigt werden. Sie können sie immer zu einem späteren Zeitpunkt bei Bedarf neu erstellen. Zumindest sollten Sie das Passwort ändern oder den Zugriffsschlüssel deaktivieren, sodass der ehemalige Benutzer nicht mehr zugreifen kann.

Natürlich ist die Bedeutung von ungenutzt unterschiedlich. Normalerweise ist damit eine Nichtverwendung von Anmeldeinformationen innerhalb eines bestimmten Zeitraums gemeint.

Suchen von ungenutzten Passwörtern

Sie können den verwenden AWS Management Console , um Informationen zur Passwortnutzung für Ihre Benutzer einzusehen. Wenn Sie über eine große Anzahl von Benutzern verfügen, können Sie über die Konsole einen Bericht mit den Anmeldeinformation herunterladen, in dem für jeden Benutzer die Information zur letztmaligen Benutzung des entsprechenden Passworts enthalten ist. Sie können auch über die AWS CLI oder die IAM-API auf die Informationen zugreifen.

So suchen Sie ungenutzte Passwörter (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 auf Users (Benutzer).
3. Fügen Sie, falls erforderlich, die Spalte Console last sign-in (Letzte Anmeldung an der Konsole) zur Benutzertabelle hinzu:
   a. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ( ).
   b. Wählen Sie unter Select visible columns (Sichtbare Spalten auswählen) die Option Console last sign-in (Letzte Anmeldung der Konsole) aus.
   c. Klicken Sie auf Confirm (Bestätigen), um zur Liste der Benutzer zurückzukehren.
4. In der Spalte Letzte Anmeldung in der Konsole wird das Datum angezeigt, an dem sich der Benutzer zuletzt AWS über die Konsole angemeldet hat. Anhand dieser Informationen können Sie Passwörter suchen, die innerhalb eines bestimmten Zeitraums nicht verwendet worden sind. In der Spalte wird für Benutzer mit Passwörtern, die sich zu keiner Zeit angemeldet haben, Never (Nie) angegeben. None (Keine) gibt Benutzer ohne Passwörter an. Passwörter, die in der letzten Zeit nicht verwendet wurden, können möglicherweise entfernt werden.

> **Important**
>
> Aufgrund eines Serviceproblems werden zwischen dem 3. Mai 2018, 22:50 Uhr PDT, und dem 23. Mai 2018, 14:08 Uhr PDT, verwendete Passwörter nicht als zuletzt verwendetes Passwort gemeldet. Dies wirkt sich auf das Datum der letzten Anmeldung aus, das in der IAM-Konsole angezeigt wird, und auf das Datum, an dem das Passwort

zuletzt verwendet wurde, im Bericht über die IAM-Anmeldeinformationen, die vom API-Vorgang zurückgegeben werden. GetUser Wenn Benutzer sich in dieser Zeit angemeldet haben, wird als letztes Verwendungsdatum des Passworts das Datum angegeben, an dem der Benutzer sich das letzte Mal vor dem 3. Mai 2018 angemeldet hat. Für Benutzer, die sich nach dem 23. Mai 2018, 14:08 Uhr PDT, angemeldet haben, wird das richtige letzte Passwortverwendungsdatum zurückgegeben.
Wenn Sie Informationen zum zuletzt verwendeten Passwort verwenden, um ungenutzte Anmeldeinformationen für das Löschen zu identifizieren, z. B. wenn Sie Benutzer löschen, bei denen Sie sich AWS in den letzten 90 Tagen nicht angemeldet haben, empfehlen wir Ihnen, Ihr Testfenster so anzupassen, dass es auch Daten nach dem 23. Mai 2018 berücksichtigt. Wenn Ihre Benutzer Zugriffstasten für den AWS programmgesteuerten Zugriff verwenden, können Sie alternativ auf die Informationen zum zuletzt verwendeten Zugriffsschlüssel zurückgreifen, da diese für alle Daten korrekt sind.

So finden Sie ungenutzte Passwörter, indem Sie den Bericht mit den Anmeldeinformationen (Konsole) herunterladen

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 auf Credential report (Anmeldeinformationsbericht).
3. Wählen Sie Download Report (Bericht herunterladen) aus, um eine CSV-Datei (Datei mit durch Kommas getrennten Werten) mit dem Namen `status_reports_<date>T<time>.csv` herunterzuladen. Die fünfte Spalte enthält die Spalte `password_last_used` mit den nachstehenden Daten:

   - N/A – Benutzer ohne zugewiesenes Passwort.
   - no_information – Benutzer, die Ihr Passwort seit dem 20. Oktober 2014, als IAM begonnen hat, das Alter des Passworts zu registrieren, nicht benutzt haben.

So ermitteln Sie ungenutzte Passwörter (AWS CLI)

Führen Sie den folgenden Befehl aus, um ungenutzte Passwörter zu ermitteln:

- `aws iam list-users` gibt eine Liste von Benutzern jeweils mit einem `PasswordLastUsed`-Wert zurück. Wenn der Wert fehlt, hat der Benutzer entweder kein Passwort oder das Passwort ist

seit dem 20. Oktober 2014, als IAM begonnen hat, das Alter des Passworts zu registrieren, nicht verwendet worden.

Um unbenutzte Passwörter zu finden (AWS API)

Rufen Sie die folgende Operation auf, um ungenutzte Passwörter zu ermitteln:

- `ListUsers` gibt eine Sammlung von Benutzern zurück, die jeweils einen `<PasswordLastUsed>`-Wert haben. Wenn der Wert fehlt, hat der Benutzer entweder kein Passwort oder das Passwort ist seit dem 20. Oktober 2014, als IAM begonnen hat, das Alter des Passworts zu registrieren, nicht verwendet worden.

Weitere Informationen über die Befehle zum Herunterladen des Berichts mit den Anmeldeinformationen finden Sie unter Abrufen von Berichten zu Anmeldeinformationen (AWS CLI).

Suchen von ungenutzten Zugriffsschlüsseln

Sie können den verwenden AWS Management Console , um Informationen zur Verwendung von Zugriffsschlüsseln für Ihre Benutzer einzusehen. Wenn Sie über eine große Anzahl von Benutzern verfügen, können Sie über die Konsole einen Bericht mit den Anmeldeinformation herunterladen, in dem für jeden Benutzer die Information zur letztmaligen Benutzung des entsprechenden Zugriffsschlüssels enthalten ist. Sie können auch über die AWS CLI oder die IAM-API auf die Informationen zugreifen.

So finden Sie ungenutzte Zugriffsschlüssel (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 auf Users (Benutzer).
3. Fügen Sie, falls erforderlich, die Spalte Access key last used (Zuletzt verwendeter Zugriffsschlüssel) der Benutzertabelle hinzu:
   a. Über der Tabelle auf der rechten Seite wählen Sie das Einstellungssymbol ( ).
   b. Wählen Sie unter Select visible columns (Sichtbare Spalten auswählen) die Option Access key last used (Zugriffsschlüssel zuletzt verwendet) aus.
   c. Klicken Sie auf Confirm (Bestätigen), um zur Liste der Benutzer zurückzukehren.

4. In der Spalte Zuletzt verwendeter Zugriffsschlüssel wird die Anzahl der Tage seit dem letzten AWS programmgesteuerten Zugriff durch den Benutzer angezeigt. Anhand dieser Informationen können Sie Zugriffsschlüssel suchen, die innerhalb eines bestimmten Zeitraums nicht verwendet worden sind. Für Benutzer ohne Zugriffsschlüssel wird in der Spalte – angezeigt. Zugriffsschlüssel, die in der letzten Zeit nicht verwendet wurden, können entfernt werden.

So finden Sie ungenutzte Zugriffsschlüssel, indem Sie den Bericht mit den Anmeldeinformationen (Konsole) herunterladen

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 auf Credential Report (Anmeldeinformationsbericht).
3. Wählen Sie Download Report (Bericht herunterladen) aus, um eine CSV-Datei (Datei mit durch Kommas getrennten Werten) mit dem Namen `status_reports_<date>T<time>.csv` herunterzuladen. Die Spalten 11 bis 13 enthalten das letzte Nutzungsdatum, die Region und die Serviceinformationen für den Zugriffsschlüssel 1. Die Spalten 16 bis 18 enthalten die gleichen Informationen für den Zugriffsschlüssel 2. Der Wert lautet N/A, wenn der Benutzer nicht über einen Zugriffsschlüssel verfügt oder der Zugriffsschlüssel seit dem 22. April 2015, als IAM begonnen hat, das Alter des Zugriffsschlüssels zu registrieren, nicht verwendet worden ist.

So ermitteln Sie ungenutzte Zugriffsschlüssel (AWS CLI)

Führen Sie die folgenden Befehle aus, um ungenutzte Zugriffsschlüssel zu ermitteln:

- `aws iam list-access-keys` gibt Informationen zu den Zugriffsschlüssel eines Benutzers einschließlich `AccessKeyID` zurück.
- `aws iam get-access-key-last-used` nimmt eine Zugriffsschlüssel-ID und gibt eine Ausgabe zurück, die das `LastUsedDate`, die `Region`, in der der Zugriffsschlüssel zuletzt verwendet wurde, und den `ServiceName` des letzten angeforderten Services enthält. Wenn `LastUsedDate` fehlt, ist der Zugriffsschlüssel seit dem 22. April 2015 – der Termin, ab dem IAM das Alter von Zugriffsschlüsseln registriert – nicht verwendet worden.

Um ungenutzte Zugriffsschlüssel (AWS API) zu finden

Rufen Sie die folgenden Operationen auf, um ungenutzte Zugriffsschlüssel zu ermitteln:

- `ListAccessKeys` gibt eine Liste der `AccessKeyID`-Werte für die Zugriffsschlüssel zurück, die dem angegebenen Benutzer zugeordnet sind.
- `GetAccessKeyLastUsed` nimmt eine Zugriffsschlüssel-ID und gibt eine Zusammenstellung von Werten zurück. Darin sind das `LastUsedDate`, die `Region`, in der der Zugriffsschlüssel zuletzt verwendet wurde, und der `ServiceName` des zuletzt angeforderten Services enthalten. Wenn der Wert fehlt, hat entweder der Benutzer keinen Zugriffsschlüssel oder der Zugriffsschlüssel ist seit dem 22. April 2015 – der Termin, ab dem IAM das Alter von Zugriffsschlüsseln registriert – nicht verwendet worden.

Weitere Informationen über die Befehle zum Herunterladen des Berichts mit den Anmeldeinformationen finden Sie unter Abrufen von Berichten zu Anmeldeinformationen (AWS CLI).

Abrufen von Berichten zu Anmeldeinformationen für Ihr AWS-Konto

Sie können einen Bericht zu Anmeldeinformationen erstellen und herunterladen. In diesem Bericht sind alle Benutzer unter Ihrem Konto mit dem Status ihrer verschiedenen Anmeldeinformationen aufgeführt (z. B. Passwörter, Zugriffsschlüssel und MFA-Geräte). Sie können einen Anmeldeinformationsbericht über die AWS Management ConsoleAWS SDKs und Befehlszeilentools oder die IAM-API abrufen.

Berichte zu Anmeldeinformationen können zu Audit- und Compliance-Zwecken verwendet werden. Anhand des Berichts können Sie prüfen, wie sich die Lebensdaueranforderungen von Anmeldeinformationen wie Passwort und Zugriffsschlüssel-Aktualisierung auswirken. Stellen Sie den Bericht einem externen Prüfer bereit oder gewähren Sie einem Prüfer die notwendigen Berechtigungen zum Herunterladen des Berichts.

Bei Bedarf können Sie alle vier Stunden einen Bericht zu Anmeldeinformationen erstellen. Wenn Sie einen Bericht anfordern, prüft IAM zunächst, ob innerhalb der letzten vier Stunden ein Bericht für den generiert AWS-Konto wurde. Wenn innerhalb dieses Zeitraums bereits ein Bericht erstellt wurde, wird dieser heruntergeladen. Wenn der aktuelle Bericht für das Konto älter ist als vier Stunden oder wenn noch keine Berichte für das Konto vorliegen, erstellt IAM einen neuen Bericht und lädt diesen herunter.

Themen

- Erforderliche Berechtigungen
- Erläuterung des Berichtsformats
- Abrufen von Berichten zu Anmeldeinformationen (Konsole)

- Abrufen von Berichten zu Anmeldeinformationen (AWS CLI)
- Abrufen von Berichten über Anmeldeinformationen (AWS API)

Erforderliche Berechtigungen

Die folgenden Berechtigungen werden zum Erstellen und Herunterladen von Berichten benötigt:

- So erstellen Sie einen Bericht zu Anmeldeinformationen: `iam:GenerateCredentialReport`
- So laden Sie den Bericht herunter: `iam:GetCredentialReport`

Erläuterung des Berichtsformats

Berichte zu Anmeldeinformationen liegen im CSV-Format vor. Sie können CSV-Dateien zur Analyse mit einer Tabellensoftware öffnen oder eine eigene Anwendung erstellen, um die CSV-Dateien programmgesteuert auszulesen und benutzerdefinierte Analysen auszuführen.

Die CSV-Datei enthält die folgenden Spalten:

user

Der Anzeigename des Benutzers

arn

Der Amazon-Ressourcenname (ARN) des Benutzers. Weitere Informationen zu ARNs finden Sie unter IAM-ARNs.

user_creation_time

Erstellungsdatum und -uhrzeit des Benutzers im ISO-8601-Format.

password_enabled

Wenn der Benutzer ein Passwort besitzt, ist dieser Wert `TRUE`, Andernfalls ist es `FALSE` der Fall. Der Wert für Root-Benutzer des AWS-Kontos ist immer. `not_supported`

password_last_used

Datum und Uhrzeit, an dem das Passwort des Benutzers Root-Benutzer des AWS-Kontos oder zuletzt für die Anmeldung auf einer AWS Website verwendet wurde, im Datums-/Uhrzeitformat nach ISO 8601. AWS Websites, die den Zeitpunkt der letzten Anmeldung eines Benutzers erfassen AWS Management Console, sind die AWS Diskussionsforen und der AWS Marketplace.

Wenn ein Passwort innerhalb von 5 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet.

- Der Wert in diesem Feld ist in folgenden Fällen `no_information`:
  - Das Passwort des Benutzers wurde noch nie verwendet.
  - Dem Passwort sind keine Anmeldedaten zugeordnet. Das ist beispielsweise der Fall, wenn das Passwort eines Benutzers nach Beginn der Erfassung dieser Informationen am 20. Oktober 2014 in IAM noch nicht verwendet wurde.
- Der Wert in diesem Feld ist `N/A` (nicht anwendbar), wenn der Benutzer nicht über ein Passwort verfügt.

Important

Aufgrund eines Serviceproblems werden zwischen dem 3. Mai 2018, 22:50 Uhr PDT, und dem 23. Mai 2018, 14:08 Uhr PDT, verwendete Passwörter nicht als zuletzt verwendetes Passwort gemeldet. Dies wirkt sich auf das Datum der letzten Anmeldung aus, das in der IAM-Konsole angezeigt wird, und auf das Datum, an dem das Passwort zuletzt verwendet wurde, im Bericht über die IAM-Anmeldeinformationen, die vom API-Vorgang zurückgegeben werden. GetUser Wenn Benutzer sich in dieser Zeit angemeldet haben, wird als letztes Verwendungsdatum des Passworts das Datum angegeben, an dem der Benutzer sich das letzte Mal vor dem 3. Mai 2018 angemeldet hat. Für Benutzer, die sich nach dem 23. Mai 2018, 14:08 Uhr PDT, angemeldet haben, wird das richtige letzte Passwortverwendungsdatum zurückgegeben.
Wenn Sie Informationen zum zuletzt verwendeten Passwort verwenden, um ungenutzte Anmeldeinformationen für das Löschen zu identifizieren, z. B. wenn Sie Benutzer löschen, bei denen Sie sich AWS in den letzten 90 Tagen nicht angemeldet haben, empfehlen wir Ihnen, Ihr Testfenster so anzupassen, dass es auch Daten nach dem 23. Mai 2018 berücksichtigt. Wenn Ihre Benutzer Zugriffstasten für den AWS programmgesteuerten Zugriff verwenden, können Sie alternativ auf die Informationen zum zuletzt verwendeten Zugriffsschlüssel zurückgreifen, da diese für alle Daten korrekt sind.

password_last_changed

Datum und Uhrzeit der letzten Passwortänderung im ISO-8601-Format. Wenn der Benutzer nicht über ein Passwort verfügt, ist der Wert in diesem Feld `N/A` (nicht anwendbar). Der Wert für AWS-Konto (Stamm) ist immer. `not_supported`

password_next_rotation

Wenn das Konto über eine Passwortrichtlinie verfügt, die eine Passwortrotation erfordert, enthält dieses Feld das Datum und die Uhrzeit im ISO 8601-Format, zu dem/der der Benutzer ein neues Passwort festlegen muss. Der Wert für AWS-Konto (Wurzel) ist immer`not_supported`.

mfa_active

Wenn ein Multi-Factor Authentication-Gerät (MFA-Gerät) für den Benutzer aktiviert wurde, ist dieser Wert `TRUE`, andernfalls `FALSE`.

access_key_1_active

Wenn der Benutzer über einen Zugriffsschlüssel verfügt und der Status des Zugriffsschlüssels `Active` ist, ist dieser Wert `TRUE`, andernfalls `FALSE`.

access_key_1_last_rotated

Datum und Uhrzeit im ISO 8601-Format, an dem der Zugriffsschlüssel des Benutzers erstellt oder zuletzt geändert wurde. Wenn der Benutzer nicht über einen aktiven Zugriffsschlüssel verfügt, ist der Wert in diesem Feld `N/A` (nicht anwendbar).

access_key_1_last_used_date

Datum und Uhrzeit im ISO 8601-Format, an dem der Zugriffsschlüssel des Benutzers zuletzt für eine AWS -API-Anforderung verwendet wurde. Wenn ein Zugriffsschlüssel innerhalb von 15 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet.

Der Wert in diesem Feld ist in folgenden Fällen `N/A` (nicht anwendbar):

- Der Benutzer verfügt nicht über einen Zugriffsschlüssel.
- Der Zugriffsschlüssel wurde noch nie verwendet.
- Der Zugriffsschlüssel wurde seit Beginn der Erfassung dieser Informationen in IAM am 22. April 2015 noch nicht verwendet.

access_key_1_last_used_region

Die AWS -Region, in der der Zugriffsschlüssel zuletzt verwendet wurde. Wenn ein Zugriffsschlüssel innerhalb von 15 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet.

Der Wert in diesem Feld ist in folgenden Fällen `N/A` (nicht anwendbar):

- Der Benutzer verfügt nicht über einen Zugriffsschlüssel.

- Der Zugriffsschlüssel wurde noch nie verwendet.
- Der Zugriffsschlüssel wurde zuletzt vor Beginn der Erfassung dieser Informationen in IAM am 22. April 2015 verwendet.
- Der zuletzt verwendete Service ist nicht regionsspezifisch, z. B. Amazon S3.

access_key_1_last_used_service

Der AWS Dienst, auf den zuletzt mit dem Zugriffsschlüssel zugegriffen wurde. Für den Wert in diesem Feld wird der `s3`Namespace des Services verwendet, z. B. für Amazon S3; und `ec2` für Amazon EC2. Wenn ein Zugriffsschlüssel innerhalb von 15 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet.

Der Wert in diesem Feld ist in folgenden Fällen `N/A` (nicht anwendbar):

- Der Benutzer verfügt nicht über einen Zugriffsschlüssel.
- Der Zugriffsschlüssel wurde noch nie verwendet.
- Der Zugriffsschlüssel wurde zuletzt vor Beginn der Erfassung dieser Informationen in IAM am 22. April 2015 verwendet.

access_key_2_active

Wenn der Benutzer über einen zweiten Zugriffsschlüssel verfügt und der Status des zweiten Zugriffsschlüssels `Active` ist, ist dieser Wert `TRUE`, andernfalls `FALSE`.

> **Note**
>
> Benutzer können über bis zu zwei Zugriffsschlüssel verfügen, um die Rotation zu vereinfachen, indem sie zuerst den Schlüssel aktualisieren und dann den vorherigen Schlüssel löschen. Weitere Informationen zum Aktualisieren der Zugriffsschlüssel finden Sie unter Aktualisierung der Zugriffsschlüssel

access_key_2_last_rotated

Datum und Uhrzeit im ISO 8601-Format, an dem der zweite Zugriffsschlüssel des Benutzers erstellt oder zuletzt aktualisiert wurde. Wenn der Benutzer nicht über einen zweiten aktiven Zugriffsschlüssel verfügt, ist der Wert in diesem Feld `N/A` (nicht anwendbar).

access_key_2_last_used_date

Datum und Uhrzeit im Datums-/Uhrzeitformat nach ISO 8601, an dem der zweite Zugriffsschlüssel des Benutzers zuletzt zum Signieren einer AWS API-Anfrage verwendet wurde. Wenn ein

Zugriffsschlüssel innerhalb von 15 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet.

Der Wert in diesem Feld ist in folgenden Fällen `N/A` (nicht anwendbar):

- Der Benutzer verfügt nicht über einen zweiten Zugriffsschlüssel.
- Der zweite Zugriffschlüssel des Benutzers wurde noch nie verwendet.
- Der zweite Zugriffsschlüssel des Benutzers wurde zuletzt vor Beginn der Erfassung dieser Informationen in IAM am 22. April 2015 verwendet.

access_key_2_last_used_region

Die AWS -Region, in der der zweite Zugriffsschlüssel des Benutzers zuletzt verwendet wurde. Wenn ein Zugriffsschlüssel innerhalb von 15 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet. Der Wert in diesem Feld ist in folgenden Fällen `N/A` (nicht anwendbar):

- Der Benutzer verfügt nicht über einen zweiten Zugriffsschlüssel.
- Der zweite Zugriffschlüssel des Benutzers wurde noch nie verwendet.
- Der zweite Zugriffsschlüssel des Benutzers wurde zuletzt vor Beginn der Erfassung dieser Informationen in IAM am 22. April 2015 verwendet.
- Der zuletzt verwendete Service ist nicht regionsspezifisch, z. B. Amazon S3.

access_key_2_last_used_service

Der AWS Dienst, auf den zuletzt mit dem zweiten Zugriffsschlüssel des Benutzers zugegriffen wurde. Für den Wert in diesem Feld wird der `s3`Namespace des Services verwendet, z. B. für Amazon S3; und `ec2` für Amazon EC2. Wenn ein Zugriffsschlüssel innerhalb von 15 Minuten mehrfach verwendet wird, wird in diesem Feld nur die erste Verwendung aufgezeichnet. Der Wert in diesem Feld ist in folgenden Fällen `N/A` (nicht anwendbar):

- Der Benutzer verfügt nicht über einen zweiten Zugriffsschlüssel.
- Der zweite Zugriffschlüssel des Benutzers wurde noch nie verwendet.
- Der zweite Zugriffsschlüssel des Benutzers wurde zuletzt vor Beginn der Erfassung dieser Informationen in IAM am 22. April 2015 verwendet.

cert_1_active

Wenn der Benutzer über ein X.509-Signaturzertifikat verfügt und der Status des Zertifikats `Active` ist, ist dieser Wert `TRUE`, andernfalls `FALSE`.

cert_1_last_rotated

Datum und Uhrzeit im ISO 8601-Format, an dem das Signaturzertifikat des Benutzers erstellt oder zuletzt geändert wurde. Wenn der Benutzer nicht über ein aktives Signaturzertifikat verfügt, ist der Wert in diesem Feld `N/A` (nicht anwendbar).

cert_2_active

Wenn der Benutzer über ein zweites X.509-Signaturzertifikat verfügt und der Status des Zertifikats `Active` ist, ist dieser Wert `TRUE`, andernfalls `FALSE`.

> **Note**
>
> Benutzer können für einen einfacheren Zertifikataustausch über bis zu zwei X.509-Signaturzertifikate verfügen.

cert_2_last_rotated

Datum und Uhrzeit im ISO 8601-Format, an dem das zweite Signaturzertifikat des Benutzers erstellt oder zuletzt geändert wurde. Wenn der Benutzer nicht über ein zweites aktives Signaturzertifikat verfügt, ist der Wert in diesem Feld `N/A` (nicht anwendbar).

Abrufen von Berichten zu Anmeldeinformationen (Konsole)

Sie können den verwenden AWS Management Console , um einen Bericht mit Anmeldeinformationen als Datei mit kommagetrennten Werten (CSV) herunterzuladen.

So laden Sie einen Bericht zu Anmeldeinformationen herunter (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 auf Credential report (Anmeldeinformationsbericht).
3. Wählen Sie Download Report (Bericht herunterladen).

Abrufen von Berichten zu Anmeldeinformationen (AWS CLI)

So laden Sie einen Bericht zu Anmeldedaten herunter (AWS CLI)

1. Generieren Sie einen Bericht über Anmeldeinformationen. AWS speichert einen einzelnen Bericht. Wenn ein Bericht vorhanden ist, überschreibt das Generieren eines Berichts für Anmeldeinformationen den vorherigen Bericht. `aws iam generate-credential-report`
2. Zeigen Sie den letzten Bericht an, der generiert wurde: `aws iam get-credential-report`

Abrufen von Berichten über Anmeldeinformationen (AWS API)

Um einen Bericht über Anmeldeinformationen (AWS API) herunterzuladen

1. Generieren Sie einen Bericht über Anmeldeinformationen. AWS speichert einen einzelnen Bericht. Wenn ein Bericht vorhanden ist, überschreibt das Generieren eines Berichts für Anmeldeinformationen den vorherigen Bericht. `GenerateCredentialReport`
2. Zeigen Sie den letzten Bericht an, der generiert wurde: `GetCredentialReport`

Verwenden von IAM mit CodeCommit: Git-Anmeldeinformationen, SSH-Schlüsseln und AWS Zugriffsschlüsseln

CodeCommit ist ein verwalteter Versionskontrolldienst, der private Git-Repositorys in der AWS Cloud hostet. Zur Verwendung CodeCommit konfigurierst du deinen Git-Client für die Kommunikation mit CodeCommit Repositorys. Im Rahmen dieser Konfiguration stellen Sie IAM-Anmeldeinformationen bereit, mit denen Sie authentifiziert CodeCommit werden können. IAM unterstützt CodeCommit drei Arten von Anmeldeinformationen:

- Git-Anmeldeinformationen, ein von IAM generiertes Paar aus Benutzername und Passwort, mit dem Sie über HTTPS mit CodeCommit Repositorys kommunizieren können.
- SSH-Schlüssel, ein lokal generiertes öffentlich-privates key pair, das Sie Ihrem IAM-Benutzer zuordnen können, um mit Repositorys über SSH zu kommunizieren. CodeCommit
- AWS Zugriffsschlüssel, die Sie mit dem im Lieferumfang enthaltenen Credential Helper verwenden können, um mit Repositorys über HTTPS zu kommunizieren. AWS CLI CodeCommit

> **Note**
>
> Sie können keine SSH-Schlüssel oder Git-Anmeldeinformationen verwenden, um auf Repositorys in einem anderen AWS -Konto zuzugreifen. Informationen zur Konfiguration des Zugriffs auf CodeCommit Repositorys für IAM-Benutzer und -Gruppen in einem anderen AWS-Konto Verzeichnis finden Sie unter Kontenübergreifendes Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen konfigurieren im Benutzerhandbuch.AWS CodeCommit

Weitere Informationen zu jeder Option finden Sie in folgenden Abschnitten.

Verwenden Sie Git-Anmeldeinformationen und HTTPS mit CodeCommit (empfohlen)

Bei GIT-Anmeldeinformationen erstellen Sie einen statischen Benutzernamen und ein Passwort für Ihre IAM-Benutzer und verwenden dann diese Anmeldeinformationen für HTTPS-Verbindungen. Sie können diese Anmeldeinformationen mit einem Drittanbieter-Tool oder einer integrierten Entwicklungsumgebung (Integrated Development Environment, IDE) verwenden, die statische Git-Anmeldeinformationen unterstützt.

Da diese Anmeldeinformationen für alle unterstützten Betriebssysteme universell gültig und mit den meisten Systemen zur Verwaltung von Anmeldeinformationen, Entwicklungsumgebungen und anderen Tools zur Softwareentwicklung kompatibel sind, wird diese Methode bevorzugt. Sie können das Passwort für Git-Anmeldeinformationen jederzeit zurücksetzen. Sie können die Anmeldeinformationen auch deaktivieren oder löschen, wenn Sie sie nicht mehr benötigen.

> **Note**
>
> Sie können nicht Ihren eigenen Benutzernamen oder das Passwort für Git-Anmeldeinformationen wählen. IAM generiert diese Anmeldeinformationen für Sie, um sicherzustellen, dass sie den Sicherheitsstandards für Repositorys entsprechen AWS und die Repositorys in sichern. CodeCommit Sie können die Anmeldeinformationen nur einmal herunterladen, unmittelbar nachdem sie generiert worden sind. Stellen Sie sicher, dass Sie die Anmeldeinformationen an einem sicheren Ort speichern. Falls erforderlich, können Sie das Passwort jederzeit zurücksetzen, allerdings funktionieren dann alle mit dem alten Passwort konfigurierten Verbindungen nicht mehr. Sie müssen die Verbindungen mit dem neuen Passwort neu konfigurieren, bevor Sie eine Verbindung herstellen können.

Weitere Informationen finden Sie im folgenden Thema:

- Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto.
- Informationen zum Generieren und Verwenden von Git-Anmeldeinformationen finden Sie unter Für HTTPS-Benutzer CodeCommit, die Git-Anmeldeinformationen verwenden im AWS CodeCommit Benutzerhandbuch.

> **Note**
>
> Wenn Sie den Namen eines IAM-Benutzers ändern, nachdem die Git-Anmeldeinformationen generiert wurden, werden der Benutzername und die Git-Anmeldeinformationen nicht automatisch geändert. Der Benutzername und das Passwort bleiben gleich und sind weiterhin gültig.

So aktualisieren Sie servicespezifische Anmeldeinformationen

1. Erstellen Sie, zusätzlich zu den bereits verwendeten, einen zweiten Satz servicespezifischer Anmeldeinformationen.
2. Aktualisieren Sie alle Ihre Anwendungen für die Nutzung der neuen Anmeldeinformationen und vergewissern Sie sich, dass die Anwendungen funktionieren.
3. Ändern Sie den Status der ursprünglichen Anmeldeinformationen in "Inaktiv".
4. Stellen Sie sicher, dass alle Ihre Anwendungen weiterhin funktionieren.
5. Löschen Sie die inaktiven, servicespezifischen Anmeldeinformationen.

Verwenden Sie SSH-Schlüssel und SSH mit CodeCommit

Mit SSH-Verbindungen erstellen Sie öffentliche und private Schlüsseldateien auf Ihrem lokalen Computer, die Git CodeCommit verwenden und für die SSH-Authentifizierung verwenden. Verknüpfen Sie den öffentlichen Schlüssel mit Ihrem IAM-Benutzer und speichern Sie den privaten Schlüssel auf Ihrem lokalen Computer ab. Weitere Informationen finden Sie im folgenden Thema:

- Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto.

- Um einen öffentlichen SSH-Schlüssel zu erstellen und ihn mit einem IAM-Benutzer zu verknüpfen, siehe Für SSH-Verbindungen unter Linux, macOS oder Unix oder siehe Für SSH-Verbindungen unter Windows im AWS CodeCommit Benutzerhandbuch.

> **Note**
>
> Der öffentliche Schlüssel muss in ssh-rsa-Format oder PEM-Format verschlüsselt sein. Die Mindest-Bit-Länge des öffentlichen Schlüssels beträgt 2048 Bit und die Maximallänge beträgt 16384 Bit. Dies ist von der Größe der hochgeladenen Datei unabhängig. Sie können beispielsweise einen 2048-Bit-Schlüssel erstellen, und die ausgegebene PEM-Datei ist 1679 Byte lang. Wenn Sie den öffentlichen Schlüssel in einem anderen Format oder einer anderen Größe bereitstellen, wird eine Fehlermeldung angezeigt, dass das Schlüsselformat ungültig ist.

Verwende HTTPS mit dem AWS CLI Credential Helper und CodeCommit

Als Alternative zu HTTPS-Verbindungen mit Git-Anmeldeinformationen können Sie Git erlauben, eine kryptografisch signierte Version Ihrer IAM-Benutzeranmeldedaten oder Ihrer Amazon EC2 EC2-Instance-Rolle zu verwenden, wann immer Git sich authentifizieren muss, um mit AWS Repositorys zu interagieren. CodeCommit Dies ist die einzige Verbindungsmethode für CodeCommit Repositorys, für die kein IAM-Benutzer erforderlich ist. Dies ist auch die einzige Methode, die mit Verbundzugriff und temporären Anmeldeinformationen funktioniert. Weitere Informationen finden Sie im folgenden Thema:

- Weitere Informationen zum Verbundzugriff finden Sie unter Identitätsanbieter und Verbund und Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund).
- Weitere Informationen zu temporären Anmeldeinformationen finden Sie unter Temporäre IAM Sicherheitsanmeldeinformationen und Temporärer Zugriff auf Repositorys. CodeCommit

Der AWS CLI Credential Helper ist nicht mit anderen Credential Helper-Systemen wie Keychain Access oder Windows Credential Management kompatibel. Wenn Sie HTTPS-Verbindungen mit dem Credential Helper für Anmeldeinformationen konfigurieren, sind weitere Einstellungen zu beachten. Weitere Informationen finden Sie unter Für HTTPS-Verbindungen unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper oder HTTPS-Verbindungen unter Windows mit dem AWS CLI Credential Helper im AWS CodeCommit Benutzerhandbuch.

Verwendung von IAM mit Amazon Keyspaces (für Apache Cassandra)

Amazon Keyspaces (for Apache Cassandra) ist ein skalierbarer, hochverfügbarer und verwalteter Apache Cassandra-kompatibler Datenbankservice. Sie können über oder programmgesteuert auf Amazon Keyspaces zugreifen. AWS Management Console Um programmgesteuert mit servicespezifischen Anmeldeinformationen auf Amazon Keyspaces zuzugreifen, können Sie `cqlsh` oder Open-Source-Cassandra-Treiber verwenden. Servicespezifische Anmeldeinformationen beinhalten einen Benutzernamen und ein Passwort, wie sie von Cassandra für die Authentifizierung und Zugriffsverwaltung verwendet werden. Sie können maximal zwei Sätze servicespezifischer Anmeldeinformationen für jeden unterstützten Service pro Benutzer festlegen.

Um programmgesteuert mit Zugriffsschlüsseln auf Amazon Keyspaces AWS zuzugreifen, können Sie das AWS SDK, die AWS Command Line Interface (AWS CLI) oder Open-Source-Cassandra-Treiber mit dem SigV4-Plugin verwenden. Weitere Informationen finden Sie unter Programmgesteuert mit Amazon Keyspaces verbinden im Amazon Keyspaces (für Apache Cassandra)-Entwicklerhandbuch.

> **Note**
>
> Wenn Sie planen, mit Amazon Keyspaces nur über die Konsole zu interagieren, müssen Sie keine servicespezifischen Anmeldeinformationen generieren. Weitere Informationen finden Sie unter Zugreifen auf Amazon Keyspaces mithilfe der Konsole im Amazon Keyspaces (für Apache Cassandra)-Entwicklerhandbuch.

Weitere Informationen über die erforderlichen Berechtigungen für den Zugriff auf Amazon Keyspaces finden Sie unter Beispiele für identitätsbasierte Amazon Keyspaces (für Apache Cassandra) Richtlinien im Amazon Keyspaces (für Apache Cassandra) Developer Guide.

Erstellen von Amazon Keyspaces Anmeldeinformationen (Konsole)

Sie können die verwenden AWS Management Console , um Amazon Keyspaces-Anmeldeinformationen (für Apache Cassandra) für Ihre IAM-Benutzer zu generieren.

So erzeugen Sie servicespezifische Amazon Keyspaces-Anmeldeinformationen (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. Wählen Sie im Navigationsbereich Users (Benutzer) und wählen Sie dann den Namen des Benutzers, der die Anmeldeinformationen benötigt.

3. Wählen Sie auf der Registerkarte Security Credentials (Sicherheitsanmeldeinformationen) unter Credentials for Amazon Keyspaces Apache Cassandra Service, Generate credentials (Anmeldeinformationen generieren).
4. Ihre servicespezifischen Anmeldeinformationen sind nun verfügbar. Das Passwort kann nur noch zu diesem Zeitpunkt eingesehen oder heruntergeladen werden. Später kann er nicht mehr wiederhergestellt werden. Sie können Ihr Passwort jedoch jederzeit zurücksetzen. Speichern Sie den Benutzer und das Passwort an einem sicheren Ort, da Sie sie später benötigen.

Erstellen von Anmeldeinformationen für Amazon Keyspaces (AWS CLI)

Sie können die verwenden AWS CLI , um Amazon Keyspaces-Anmeldeinformationen (für Apache Cassandra) für Ihre IAM-Benutzer zu generieren.

So generieren Sie servicespezifische Amazon Keyspaces-Anmeldeinformationen (AWS CLI)

- Verwenden Sie den folgenden Befehl:
  - als ich create-service-specific-credential

Generierung von Amazon Keyspaces-Anmeldeinformationen (AWS API)

Sie können die AWS API verwenden, um Amazon Keyspaces-Anmeldeinformationen (für Apache Cassandra) für Ihre IAM-Benutzer zu generieren.

Um servicespezifische Anmeldeinformationen (AWS API) von Amazon Keyspaces zu generieren

- Führen Sie die folgende Operation aus:
  - CreateServiceSpecificCredential

Verwalten von Serverzertifikaten in IAM

Um HTTPS-Verbindungen zu Ihrer Website oder Anwendung zu aktivieren AWS, benötigen Sie ein SSL/TLS-Serverzertifikat. Für Zertifikate in einer Region, die von AWS Certificate Manager (ACM) unterstützt wird, empfehlen wir die Verwendung von für Bereitstellung und Verwaltung Ihrer Server-Zertifikate. In nicht unterstützten Regionen müssen Sie IAM für die Verwaltung der Zertifikate verwenden. Informationen darüber, welche Regionen ACM unterstützt, finden Sie unter AWS Certificate Manager -Endpunkte und -Kontingente im Allgemeine AWS-Referenz.

Für die Bereitstellung und Verwaltung von Serverzertifikaten empfehlen wir ACM zu verwenden. Mit ACM können Sie ein Zertifikat anfordern oder ein vorhandenes ACM- oder externes Zertifikat für Ressourcen bereitstellen. AWS Die von ACM bereitgestellten Zertifikate sind kostenlos und werden automatisch verlängert. In einer unterstützten Region können Sie mit ACM Serverzertifikate über die Konsole oder programmgesteuert verwalten. Weitere Informationen zu ACM finden Sie im AWS Certificate Manager -Leitfaden. Weitere Informationen zum Anfordern eines -Zertifikats finden Sie unter Anfordern eines öffentlichen Zertifikats oder Anfordern eines privaten Zertifikats AWS Certificate Manager im . Weitere Informationen zum Importieren von Zertifikaten von Drittanbietern in ACM finden Sie unter Importieren von Zertifikaten im AWS Certificate Manager -Benutzerhandbuch.

Verwenden Sie IAM als Zertifikatmanager nur für die Unterstützung von HTTPS-Verbindungen in einer Region, die nicht von ACM unterstützt wird. IAM bietet eine sichere Verschlüsselungsmethode für Ihre privaten Schlüssel und speichert die verschlüsselte Version in einem SSL-Zertifikatspeicher in IAM. IAM unterstützt die Bereitstellung von Serverzertifikaten in allen Regionen, Sie müssen Ihr Zertifikat jedoch von einem externen Anbieter beziehen, damit Sie es verwenden können. AWS ACM-Zertifikate können nicht in IAM hochgeladen werden. Außerdem ist es nicht möglich, Zertifikate auf der IAM-Konsole zu verwalten.

Weitere Informationen zum Hochladen von Drittanbieterzertifikaten in IAM finden Sie in den folgenden Themen.

Inhalt

Hochladen eines Serverzertifikats (API)AWS

Um ein Serverzertifikat in IAM hochzuladen, müssen Sie das Zertifikat bereitstellen und dessen privaten Schlüssel verwalten. Wenn das Zertifikat nicht selbstsigniert ist, müssen Sie auch eine Zertifikatskette bereitstellen. (Zum Hochladen eines selbstsignierten Zertifikats benötigen

Sie keine Zertifikatskette.) Bevor Sie ein Zertifikat hochladen, stellen Sie sicher, dass folgende Voraussetzungen erfüllt sind:

- Das Zertifikat muss zum Zeitpunkt des Hochladens gültig sein. Sie können ein Zertifikat nicht vor Beginn des Gültigkeitszeitraums (das Datum `NotBefore` des Zertifikats) oder nach Ablauf der Gültigkeit (das Datum `NotAfter` des Zertifikats) hochladen.
- Der private Schlüssel muss unverschlüsselt sein. Sie können keinen privaten Schlüssel hochladen, der durch ein Passwort oder eine Passphrase geschützt ist. Informationen zum Entschlüsseln von verschlüsselten privaten Schlüsseln finden Sie unter Fehlerbehebung.
- Das Zertifikat, der private Schlüssel und die Zertifikatskette müssen PEM-kodiert sein. Informationen zum Konvertieren dieser Elemente ins PEM-Format finden Sie unter Fehlerbehebung.

Um die IAM-API zum Hochladen eines Zertifikats zu verwenden, senden Sie eine UploadServerCertificateAnfrage. Im folgenden Beispiel wird gezeigt, wie Sie dies mit dem AWS Command Line Interface (AWS CLI) durchführen. In diesem Beispiel wird Folgendes angenommen:

- Das PEM-kodierte Zertifikat ist in einer Datei mit dem Namen `Certificate.pem` gespeichert.
- Das PEM-kodierte Zertifikatkette ist in einer Datei mit dem Namen `CertificateChain.pem` gespeichert.
- Das PEM-kodierte Zertifikat ist in einer Datei mit dem Namen `PrivateKey.pem` gespeichert.
- (Optional) Sie möchten das Serverzertifikat mit einem Schlüsselwertpaar kennzeichnen. Sie können beispielsweise den Tag-Schlüssel `Department` und den Tag-Wert `Engineering` hinzufügen, um Ihnen bei der Identifizierung und Organisation Ihrer Zertifikate zu helfen.

Um den folgenden Beispielbefehl zu verwenden, ersetzen Sie diese Dateinamen durch Ihre eigenen. *`ExampleCertificate`*Ersetzen Sie es durch einen Namen für Ihr hochgeladenes Zertifikat. Wenn Sie das Zertifikat taggen möchten, ersetzen Sie das Schlüssel-Wert-Paar *`ExampleKey`*und das *`ExampleValue`*Tag-Schlüssel-Wert-Paar durch Ihre eigenen Werte. Geben Sie den Befehl durchgehend in einer Zeile ein. Das folgende Beispiel enthält Zeilenumbrüche und zusätzliche Leerzeichen, um das Lesen zu vereinfachen.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
```

```
                                    --tags '{"Key": "ExampleKey", "Value":
"ExampleValue"}'
```

Wenn der vorherige Befehl erfolgreich ausgeführt wurde, werden Metadaten zum hochgeladenen Zertifikat einschließlich dessen Amazon-Ressourcennamen (ARN), dem Anzeigenamen, der ID, dem Ablaufdatum, Tags usw. zurückgegeben.

> **Note**
>
> Wenn Sie ein Serverzertifikat zur Verwendung mit Amazon hochladen CloudFront, müssen Sie mit der `--path` Option einen Pfad angeben. Der Pfad muss mit `/cloudfront` beginnen und mit einem Schrägstrich enden (z. B. `/cloudfront/test/`).

Um das zum Hochladen eines Zertifikats AWS Tools for Windows PowerShell zu verwenden, verwenden Sie ServerCertificatePublish-IAM.

Abrufen eines Serverzertifikats (API)AWS

Um die IAM-API zum Abrufen eines Zertifikats zu verwenden, senden Sie eine GetServerCertificateAnfrage. Das folgende Beispiel zeigt, wie Sie dies mit dem AWS CLI tun können. *ExampleCertificate*Ersetzen Sie es durch den Namen des abzurufenden Zertifikats.

```
aws iam get-server-certificate --server-certificate-name ExampleCertificate
```

Wenn der vorherige Befehl erfolgreich ausgeführt wurde, werden das Zertifikat, die Zertifikatskette (wenn eine hochgeladen wurde) sowie Metadaten zum Zertifikat zurückgegeben.

> **Note**
>
> Sie können private Schlüssel nach dem Hochladen nicht mehr aus IAM herunterladen oder abrufen.

Um das zum Abrufen eines Zertifikats AWS Tools for Windows PowerShell zu verwenden, verwenden Sie Get-IAM ServerCertificate.

Serverzertifikate auflisten (API)AWS

Um die IAM-API zum Auflisten Ihrer hochgeladenen Serverzertifikate zu verwenden, senden Sie eine ListServerCertificatesAnfrage. Das folgende Beispiel zeigt, wie Sie dies mit dem AWS CLI tun können.

```
aws iam list-server-certificates
```

Wenn der vorherige Befehl erfolgreich ausgeführt wurde, wird eine Liste mit Metadaten zu den einzelnen Zertifikaten zurückgegeben.

Verwenden Sie ServerCertificatesGet-IAM AWS Tools for Windows PowerShell, um Ihre hochgeladenen Serverzertifikate aufzulisten.

Markierung und Entfernung der Markierung von Serverzertifikaten (AWS -API)

Sie können Ihren IAM-Ressourcen Tags hinzufügen, um den Zugriff auf sie zu organisieren und zu kontrollieren. Um die IAM-API zum Markieren eines vorhandenen Serverzertifikats zu verwenden, senden Sie eine Anfrage. TagServerCertificate Das folgende Beispiel zeigt, wie Sie dies mit dem AWS CLI tun können.

```
aws iam tag-server-certificate --server-certificate-name ExampleCertificate
                               --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Wenn der vorhergehende Befehl erfolgreich ausgeführt wurde, wird keine Ausgabe zurückgegeben.

Um die IAM-API zum Aufheben der Markierung eines Serverzertifikats zu verwenden, senden Sie eine Anfrage. UntagServerCertificate Das folgende Beispiel zeigt, wie Sie dies mit dem AWS CLI tun können.

```
aws iam untag-server-certificate --server-certificate-name ExampleCertificate
                                 --tag-keys ExampleKeyName
```

Wenn der vorhergehende Befehl erfolgreich ausgeführt wurde, wird keine Ausgabe zurückgegeben.

Umbenennen eines Serverzertifikats oder Aktualisierung seines Pfads (API)AWS

Um mit der IAM-API ein Serverzertifikat umzubenennen oder seinen Pfad zu aktualisieren, senden Sie eine UpdateServerCertificateAnfrage. Das folgende Beispiel zeigt, wie Sie dies mit dem AWS CLI tun können.

Um den folgenden Beispielbefehl zu verwenden, ersetzen Sie den alten und neuen Zertifikatnamen und den Zertifikatspfad und geben Sie den Befehl durchgehend in einer Zeile ein. Das folgende Beispiel enthält Zeilenumbrüche und zusätzliche Leerzeichen, um das Lesen zu vereinfachen.

```
aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                  --new-server-certificate-name CloudFrontCertificate
                                  --new-path /cloudfront/
```

Wenn der vorherige Befehl erfolgreich ausgeführt wurde, erfolgt keine Ausgabe.

Verwenden Sie Update-IAM AWS Tools for Windows PowerShell , um ein Serverzertifikat umzubenennen oder seinen Pfad zu aktualisieren. ServerCertificate

Löschen eines Serverzertifikats (API)AWS

Um die IAM-API zum Löschen eines Serverzertifikats zu verwenden, senden Sie eine DeleteServerCertificateAnfrage. Das folgende Beispiel zeigt, wie Sie dies mit dem AWS CLI tun können.

Um den folgenden Beispielbefehl zu verwenden, *ExampleCertificate*ersetzen Sie ihn durch den Namen des zu löschenden Zertifikats.

```
aws iam delete-server-certificate --server-certificate-name ExampleCertificate
```

Wenn der vorherige Befehl erfolgreich ausgeführt wurde, erfolgt keine Ausgabe.

Um das zum Löschen eines Serverzertifikats AWS Tools for Windows PowerShell zu verwenden, verwenden Sie Remove-IAM. ServerCertificate

Fehlerbehebung

Bevor Sie ein Zertifikat in IAM hochladen können, müssen Sie sicherstellen, dass das Zertifikat, der private Schlüssel und die Zertifikatskette PEM-kodiert sind. Sie müssen außerdem dafür sorgen, dass der private Schlüssel nicht verschlüsselt ist. Sehen Sie sich die folgenden Beispiele an.

Example Beispiel für ein PEM-kodiertes Zertifikat

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Beispiel eines PEM-kodierten, unverschlüsselten privaten Schlüssels

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example PEM-kodierte Zertifikatkette

Eine Zertifikatkette enthält ein oder mehrere Zertifikate. Sie können Ihre Zertifikatdateien mit einem Texteditor, dem Kopierbefehl in Windows oder dem cat-Befehl in Linux zu einer Kette verknüpfen. Wenn Sie mehrere Zertifikate einbeziehen, muss jedes Zertifikat das vorhergehende Zertifikat zertifizieren. Dazu müssen Sie die Zertifikate verketten, darunter das Zertifikat der Stammzertifizierungsstelle als letztes.

Das folgende Beispiel enthält drei Zertifikate, Ihre Zertifikatkette enthält möglicherweise jedoch mehr oder weniger Zertifikate.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Wenn diese Elemente nicht im richtigen Format zum Hochladen in IAM vorliegen, können Sie sie mit OpenSSL ins richtige Format konvertieren.

So konvertieren Sie ein Zertifikat oder eine Zertifikatskette von DER zu PEM

Verwenden Sie den Befehl OpenSSL x509 wie im folgenden Beispiel. Ersetzen Sie im folgenden Beispielbefehl *Certificate.der* durch den Namen der Datei, die das DER-kodierte Zertifikat enthält. Ersetzen Sie *Certificate.pem* durch den gewünschten Namen der Ausgabedatei, die das PEM-codierte Zertifikat enthalten soll.

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

So konvertieren Sie einen privaten Schlüssel von DER zu PEM

Verwenden Sie den Befehl OpenSSL rsa wie im folgenden Beispiel. Ersetzen Sie im folgenden Beispielbefehl *PrivateKey.der* durch den Namen der Datei, die den DER-kodierten privaten Schlüssel enthält. Ersetzen Sie *PrivateKey.pem* durch den gewünschten Namen der Ausgabedatei, die den PEM-codierten privaten Schlüssel enthalten soll.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

So entschlüsseln Sie einen verschlüsselten privaten Schlüssel (Entfernen eines Passworts oder einer Passphrase)

Verwenden Sie den Befehl OpenSSL rsa wie im folgenden Beispiel. Um den folgenden Beispielbefehl zu verwenden, ersetzen Sie *EncryptedPrivateKey.pem* durch den Namen der Datei, die den verschlüsselten privaten Schlüssel enthält. Ersetzen Sie *PrivateKey.pem* durch den gewünschten Namen der Ausgabedatei, die den PEM-codierten unverschlüsselten privaten Schlüssel enthalten soll.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

So konvertieren Sie ein Zertifikat-Bundle von PKCS#12 (PFX) zu PEM

Verwenden Sie den Befehl OpenSSL pkcs12 wie im folgenden Beispiel. Ersetzen Sie im folgenden Beispielbefehl *CertificateBundle.p12* durch den Namen der Datei, die das PKCS#12-kodierte Zertifikat-Bundle enthält. Ersetzen Sie *CertificateBundle.pem* durch den gewünschten Namen der Ausgabedatei, die das PEM-codierte Zertifikat-Bundle enthalten soll.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

So konvertieren Sie ein Zertifikat-Bundle von PKCS#7 zu PEM

Verwenden Sie den Befehl OpenSSL pkcs7 wie im folgenden Beispiel. Ersetzen Sie im folgenden Beispielbefehl *CertificateBundle.p7b* durch den Namen der Datei, die das PKCS#7-kodierte Zertifikat-Bundle enthält. Ersetzen Sie *CertificateBundle.pem* durch den gewünschten Namen der Ausgabedatei, die das PEM-codierte Zertifikat-Bundle enthalten soll.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

IAM-Benutzergruppen

Eine IAM-Gruppe ist eine Auswahl von IAM-Benutzern. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzern angeben, was die Verwaltung der Berechtigungen für diese Benutzer erleichtert. Zum Beispiel können Sie eine Gruppe mit dem Namen Administratoren erstellen und dieser Gruppe die für Administratoren üblichen Berechtigungen erteilen. Jeder Benutzer in der Gruppe verfügt automatisch über Admins-Gruppenberechtigungen. Wenn ein neuer Benutzer Ihrer Organisation beitritt und Administratorrechte benötigt, können die entsprechenden Berechtigungen zuweisen, indem Sie den Benutzer zur Admins-Benutzergruppe hinzufügen. Wenn eine Person in Ihrem Unternehmen die Stelle wechselt, können Sie, anstatt die Berechtigungen des Benutzers zu bearbeiten, ihn aus den alten Benutzergruppen entfernen und ihn den entsprechenden neuen Benutzergruppen hinzufügen.

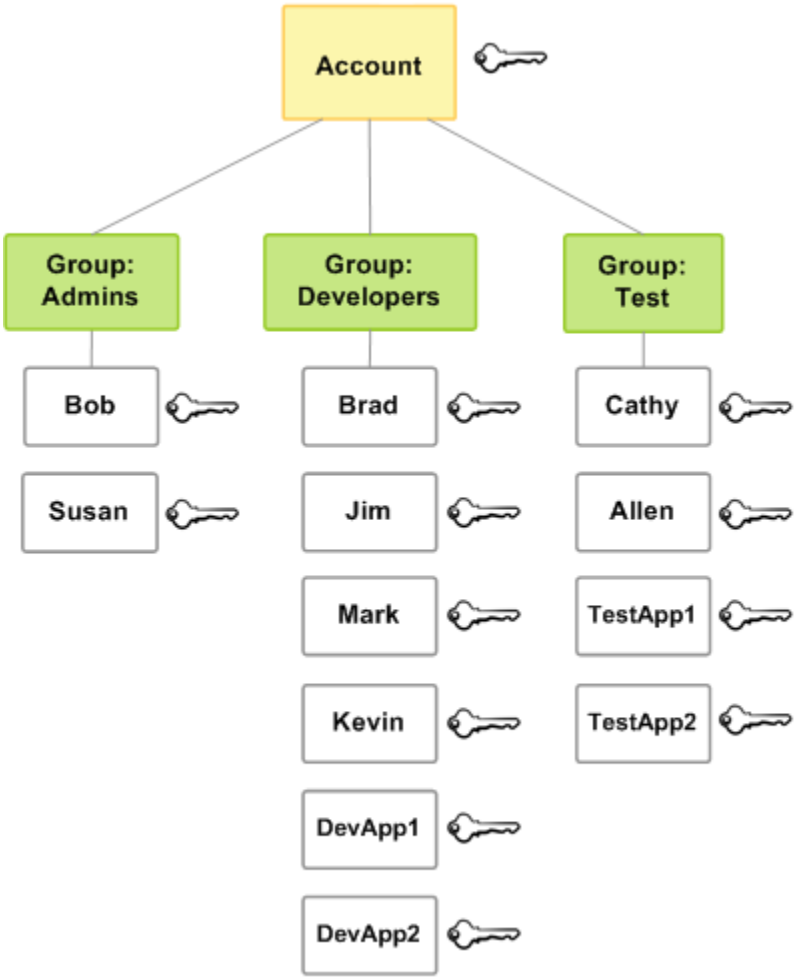
Sie können eine identitätsbasierte Richtlinie an eine Benutzergruppe anfügen, so dass alle Benutzer in der Benutzergruppe die Berechtigungen der Richtlinie erhalten. Sie können eine Benutzergruppe nicht als `Principal` in einer Richtlinie (z. B. einer ressourcenbasierten Richtlinie) identifizieren, da sich Gruppen auf Berechtigungen und nicht auf die Authentifizierung beziehen, während Prinzipale authentifizierte IAM-Entitäten sind. Weitere Informationen zu Richtlinientypen finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien.

Im Folgenden finden Sie einige wichtige Merkmale von Benutzergruppen:

- Eine Gruppe kann viele Benutzer enthalten und ein Benutzer kann mehreren Gruppen angehören.
- Gruppen können nicht verschachtelt werden. Sie können nur Benutzer, nicht andere Gruppen enthalten.
- Es gibt keine Standard-Benutzergruppe, die automatisch alle Benutzer im AWS-Konto-Konto enthält. Wenn Sie eine solche Gruppe haben möchten, müssen Sie sie erstellen und ihr jeden neuen Benutzer zuweisen.
- Die Anzahl und Größe der IAM-Ressourcen in einer AWS-Konto, z. B. die Anzahl der Gruppen und die Anzahl der Gruppen, denen ein Benutzer angehören kann, sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Die folgende Abbildung zeigt ein einfaches Beispiel für ein kleines Unternehmen. Der Unternehmenseigentümer erstellt eine `Admins`-Gruppe für Benutzer, um andere Benutzer zu

erstellen und zu verwalten, während das Unternehmen wächst. Die `Admins`-Benutzergruppe erstellt eine `Developers`-Benutzergruppe und eine `Test`-Benutzergruppe. Jede dieser Benutzergruppen besteht aus Benutzern (Menschen und Anwendungen), die mit AWS (Jim, Brad, DevApp 1 usw.) interagieren. Jeder Benutzer verfügt über individuelle Anmeldeinformationen. In diesem Beispiel gehört jeder Benutzer zu einer einzelnen Gruppe. Benutzer können jedoch mehreren Gruppen angehören.



IAM-Benutzergruppen erstellen

> **Note**
>
> Als bewährte Methode empfehlen wir, dass menschliche Benutzer für den Zugriff mit temporären Anmeldeinformationen einen Verbund mit einem Identitätsanbieter AWS verwenden müssen. Wenn Sie die bewährten Methoden befolgen, verwalten Sie keine IAM-Benutzer und -Gruppen. Stattdessen werden Ihre Benutzer und Gruppen außerhalb

von AWS Ressourcen verwaltet AWS und können als föderierte Identität darauf zugreifen. Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter, dem AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe von Anmeldeinformationen, die über eine Identitätsquelle bereitgestellt wurden, auf AWS Dienste zugreift. Verbundidentitäten verwenden die von ihrem Identitätsanbieter definierten Gruppen. Wenn Sie dies verwenden AWS IAM Identity Center, finden Sie unter Identitäten in IAM Identity Center verwalten im AWS IAM Identity Center Benutzerhandbuch Informationen zum Erstellen von Benutzern und Gruppen in IAM Identity Center.

Zum Einrichten einer Gruppe müssen Sie die Gruppe erstellen. Anschließend weisen Sie der Gruppe Berechtigungen basierend auf der Arbeit zu, die von den Benutzern in der Gruppe ausgeführt werden soll. Fügen Sie abschließend der Gruppe Benutzer hinzu.

Weitere Informationen zu den Berechtigungen, die Sie zum Erstellen eines Benutzers benötigen, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

So erstellen Sie eine IAM-Gruppe und fügen Richtlinien an (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. Wählen Sie im Navigationsbereich Groups (Gruppen) und dann Gruppe erstellen.
3. Geben Sie unter Benutzergruppenname den Namen der Gruppe ein.

   > **Note**
   >
   > Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS. Gruppennamen können eine Kombination aus bis zu 128 Buchstaben, Ziffern und den folgenden Zeichen enthalten: Plus (+), Gleichheitszeichen (=), Komma (,), Punkt (.), at-Zeichen (@), Unterstrich (_) und Bindestrich (-). Namen müssen innerhalb eines Kontos eindeutig sein. Es wird hierbei nicht zwischen Groß- und Kleinschreibung unterschieden. Beispielsweise können Sie keine Gruppen erstellen, die **`ADMINS`** bzw. **`admins`** heißen.

4. Aktivieren Sie in der Liste der Benutzer das Kontrollkästchen für jeden Benutzer, den Sie der Gruppe hinzufügen möchten.

5. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen für die jeweilige Richtlinie, die Sie auf alle Mitglieder der Gruppe anwenden möchten.
6. Wählen Sie Create group (Gruppe erstellen) aus.

Um IAM-Benutzergruppen (AWS CLI oder AWS APIs) zu erstellen

Nutzen Sie einen der Folgenden:

- AWS CLI: aws iam create-group
- AWS API: CreateGroup

Verwalten von IAM-Gruppen

Amazon Web Services bietet mehrere Tools for die Verwaltung von IAM-Benutzergruppen. Weitere Informationen zu den Berechtigungen, die Sie zum Erstellen und Entfernen von Benutzern in einer Gruppe benötigen, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

Themen

- Auflisten von IAM-Gruppen
- Hinzufügen und Entfernen von Benutzern in einer IAM-Gruppe
- Zuordnen einer Richtlinie zu einer IAM-Gruppe
- Umbenennen einer IAM-Gruppe
- Löschen einer IAM-Benutzergruppe

Auflisten von IAM-Gruppen

Sie können alle Gruppen in Ihrem Konto, die Benutzer innerhalb einer Gruppe sowie die Gruppen, denen ein Benutzer angehört, auflisten. Wenn Sie die AWS API AWS CLI oder verwenden, können Sie alle Benutzergruppen mit einem bestimmten Pfadpräfix auflisten.

So listen Sie alle Gruppen in Ihrem Konto auf

Führen Sie eine der folgenden Aktionen aus:

- AWS Management Console: Klicken Sie im Navigationsbereich auf Benutzergruppen.
- AWS CLI: aws iam list-groups
- AWS API: ListGroups

So listen Sie die Benutzer in einer bestimmten Gruppe auf

Führen Sie eine der folgenden Aktionen aus:

- AWS Management Console: Wählen Sie im Navigationsbereich Benutzergruppen, den Namen der Gruppe und anschließend die Registerkarte Benutzer.
- AWS CLI: aws iam get-group
- AWS API: GetGroup

So listen Sie alle Gruppen auf, denen ein Benutzer angehört

Führen Sie eine der folgenden Aktionen aus:

- AWS Management Console: Wählen Sie im Navigationsbereich Benutzer, den Benutzernamen und anschließend die Registerkarte Gruppen.
- AWS CLI: wie ich list-groups-for-user
- AWS API: ListGroupsForUser

Hinzufügen und Entfernen von Benutzern in einer IAM-Gruppe

Verwenden Sie Gruppen, um dieselben Berechtigungsrichtlinien für mehrere Benutzer gleichzeitig anzuwenden. Sie können dann Benutzer zu einer IAM-Gruppe hinzufügen oder aus dieser entfernen. Dies ist nützlich, wenn neue Mitarbeiter in Ihrem Unternehmen eingestellt werden oder dieses wieder verlassen.

Anzeigen des Richtlinienzugriffs

Bevor Sie die Berechtigungen für eine Richtlinie ändern, sollten Sie deren kürzliche Aktivität auf Serviceebene überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Hinzufügen oder Entfernen eines Benutzers in einer Gruppe (Konsole)

Sie können den verwenden AWS Management Console , um einen Benutzer zu einer Benutzergruppe hinzuzufügen oder aus dieser zu entfernen.

So fügen Sie einen Benutzer einer IAM-Benutzergruppe (Konsole) hinzu

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 die Option Benutzergruppen und dann den Namen der Gruppe aus.
3. Wählen Sie die Registerkarte Users und anschließend die Option Add Users to Group. Markieren Sie das Kontrollkästchen neben den Benutzern, die Sie hinzufügen möchten.
4. Wählen Sie Add Users (Benutzer hinzufügen).

So entfernen Sie einen Benutzer aus einer IAM-Gruppe (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. Wählen Sie im Navigationsbereich die Option Benutzergruppen und dann den Namen der Gruppe aus.
3. Wählen Sie die Registerkarte Users. Aktivieren Sie das Kontrollkästchen neben den Benutzern, die Sie entfernen möchten, und wählen Sie dann Benutzer entfernen.

Hinzufügen oder Entfernen eines Benutzers in einer Gruppe (AWS CLI)

Sie können den verwenden AWS CLI , um einen Benutzer einer Benutzergruppe hinzuzufügen oder aus dieser zu entfernen.

So fügen Sie einen Benutzer einer IAM-Gruppe (AWS CLI) hinzu

- Verwenden Sie den folgenden Befehl:
  - war ich add-user-to-group

So entfernen Sie einen Benutzer aus einer IAM-Gruppe (AWS CLI)

- Verwenden Sie den folgenden Befehl:
  - war ich remove-user-from-group

Einen Benutzer zu einer Benutzergruppe hinzufügen oder entfernen (AWS API)

Sie können die AWS API verwenden, um einen Benutzer zu einer Benutzergruppe hinzuzufügen oder zu entfernen.

Um einen Benutzer zu einer IAM-Gruppe (AWS API) hinzuzufügen

- Führen Sie die folgende Operation aus:
  - AddUserToGroup

Um einen Benutzer aus einer IAM-Benutzergruppe (AWS API) zu entfernen

- Führen Sie die folgende Operation aus:
  - RemoveUserFromGroup

Zuordnen einer Richtlinie zu einer IAM-Gruppe

Sie können einer Benutzergruppe eine AWS verwaltete Richtlinie — d. h. eine von vordefinierte Richtlinie AWS— zuordnen, wie in den folgenden Schritten erklärt. Um eine vom Kunden verwaltete Richtlinie anzuhängen, d. h. eine Richtlinie mit von Ihnen erstellten benutzerdefinierten Berechtigungen, müssen Sie die Richtlinie zunächst erstellen. Weitere Informationen zum Erstellen von kundenverwalteten Richtlinien finden Sie unter Erstellen von IAM-Richtlinien.

Weitere Informationen zu Berechtigungen und Richtlinien finden Sie unter Zugriffsmanagement für AWS Ressourcen.

So verknüpfen Sie eine Richtlinie mit einer Benutzergruppe (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. Wählen Sie im Navigationsbereich die Option Benutzergruppen und dann den Namen der Gruppe aus.
3. Wählen Sie die Registerkarte Berechtigungen.
4. Wählen Sie Berechtigungen hinzufügen und anschließend Richtlinien anhängen aus.
5. Die aktuellen Richtlinien, die der Benutzergruppe zugeordnet sind, werden in der Liste Aktuelle Berechtigungsrichtlinien angezeigt. Aktivieren Sie in der Liste der Richtlinien für andere

Berechtigungen das Kontrollkästchen neben den Namen der zuzuordnenden Richtlinien. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.

6. Wählen Sie die Richtlinie aus, die Sie an Ihre IAM-Benutzergruppe anhängen möchten, und wählen Sie Richtlinien anhängen aus.

Um eine Richtlinie an eine Benutzergruppe (AWS CLI oder AWS API) anzuhängen

Führen Sie eine der folgenden Aufgaben aus:

- AWS CLI: war ich attach-group-policy
- AWS API: AttachGroupPolicy

Umbenennen einer IAM-Gruppe

Wenn Sie den Namen oder den Pfad einer Gruppe ändern, geschieht Folgendes:

- Alle Richtlinien, die der Gruppe zugewiesen sind, verbleiben bei ihr unter dem neuen Namen.
- Alle der Gruppe zugeordneten Benutzer bleiben unter dem neuen Namen erhalten.
- Die eindeutige ID der Gruppe bleibt unverändert. Weitere Informationen zu eindeutigen IDs finden Sie unter Eindeutige Bezeichner.

In IAM werden die Richtlinien, die auf die Gruppe als Ressource verweisen, nicht automatisch mit dem neuen Namen aktualisiert. Dies müssen Sie manuell vornehmen. Daher müssen Sie beim Umbenennen einer Benutzergruppe vorsichtig vorgehen. Bevor Sie die Gruppe umbenennen, müssen Sie manuell alle Richtlinien prüfen, um diejenigen Richtlinien zu ermitteln, in denen der Name dieser Gruppe erwähnt wird. Angenommen, Bob ist der Leiter der Testabteilung der Organisation. Bob hat eine Richtlinie an seine IAM-Benutzer-Entity angefügt, mit der er Benutzer in die Testgruppe einfügen und aus dieser entfernen kann. Wenn ein Administrator den Namen der Gruppe (oder den Gruppenpfad) ändert, muss der Administrator auch die Bob zugeordnete Richtlinie an den neuen Namen oder Pfad anpassen. Andernfalls kann Bob keine Benutzer mehr zu der Gruppe hinzufügen oder daraus entfernen.

So finden Sie Richtlinien, die auf eine Gruppe als eine Ressource verweisen:

1. Wählen Sie im Navigationsbereich der IAM-Konsole Policies (Richtlinien).
2. Sortieren Sie nach der Spalte Typ, um Ihre vom Kunden verwalteten benutzerdefinierten Richtlinien zu finden.

3. Wählen Sie Edit policy (Richtlinie bearbeiten), um den Namen der Gruppe in der Richtlinie zu ändern.
4. Klicken Sie auf die Registerkarte Permissions (Berechtigungen) und anschließend auf Summary (Übersicht).
5. Wählen Sie IAM aus der Liste der Services aus, falls vorhanden.
6. Suchen Sie nach dem Namen Ihrer Gruppe in der Spalte Resource (Ressource).
7. Wählen Sie Edit (Bearbeiten), um den Namen der Gruppe in der Richtlinie zu ändern.

So ändern Sie den Namen einer IAM-Gruppe

Führen Sie eine der folgenden Aktionen aus:

- AWS Management Console: Wählen Sie im Navigationsbereich die Option Benutzergruppen und dann den Gruppennamen aus. Wählen Sie Edit. Geben Sie den neuen Benutzergruppennamen ein, und wählen Sie dann Speichern Sie die Änderungen.
- AWS CLI: aws iam update-group
- AWS API: UpdateGroup

Löschen einer IAM-Benutzergruppe

Wenn Sie eine Benutzergruppe in der löschen AWS Management Console, entfernt die Konsole automatisch alle Gruppenmitglieder, trennt alle angehängten verwalteten Richtlinien und löscht alle Inline-Richtlinien. Da IAM jedoch Richtlinien, die sich auf die Benutzergruppe als Ressource beziehen, nicht automatisch löscht, müssen Sie vorsichtig sein, wenn Sie eine Benutzergruppe löschen. Bevor Sie Ihre Gruppe löschen, müssen Sie manuell alle Richtlinien prüfen, um diejenigen Richtlinien zu ermitteln, in denen der Name dieser Gruppe erwähnt wird. John, der Leiter des Testteams, verfügt beispielsweise über eine Richtlinie, die an seine IAM-Benutzerentität angefügt ist und ihm das Hinzufügen und Entfernen von Benutzern aus der Test-Benutzergruppe erlaubt. Wenn ein Administrator die Gruppe löscht, muss er auch die an John angefügte Richtlinie löschen. Andernfalls bleiben, wenn der Administrator die gelöschte Gruppe neu erstellt und ihr denselben Namen gibt, die Berechtigungen von John bestehen, auch wenn er das Test-Team verlassen hat.

So finden Sie Richtlinien, die auf eine Gruppe als eine Ressource verweisen

1. Wählen Sie im Navigationsbereich der IAM-Konsole Policies (Richtlinien).

2. Sortieren Sie nach der Spalte Typ, um Ihre vom Kunden verwalteten benutzerdefinierten Richtlinien zu finden.
3. Wählen Sie den Richtliniennamen der zu löschenden Richtlinie.
4. Klicken Sie auf die Registerkarte Permissions (Berechtigungen) und anschließend auf Summary (Übersicht).
5. Wählen Sie IAM aus der Liste der Services aus, falls vorhanden.
6. Suchen Sie nach dem Namen Ihrer Gruppe in der Spalte Resource (Ressource).
7. Wählen Sie Delete (Löschen), um die Richtlinie zu löschen.
8. Geben Sie den Richtliniennamen ein, um das Löschen der Richtlinie zu bestätigen, und wählen Sie Delete (Löschen) aus.

Wenn Sie dagegen Tools für Windows oder AWS API verwenden PowerShell, um eine Benutzergruppe zu löschen, müssen Sie zuerst die Benutzer in der Gruppe entfernen. AWS CLI Löschen Sie dann alle in die Gruppe einbetteten Inline-Richtlinien. Trennen Sie dann alle an die Gruppe angefügten verwalteten Richtlinien. Erst dann können Sie die Gruppe selbst löschen.

Löschen einer IAM-Gruppe (Konsole)

Sie können eine IAM-Benutzergruppe aus dem AWS Management Console löschen.

So löschen Sie eine IAM-Gruppe (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 auf Benutzergruppen.
3. Aktivieren Sie in der Liste der Gruppen das Kontrollkästchen neben dem Namen der zu löschenden Gruppe. Über das Suchfeld können Sie die Liste der Benutzergruppen nach Typ, Berechtigungen und Benutzergruppenname filtern.
4. Wählen Sie Löschen aus.
5. Wenn Sie eine einzelne Benutzergruppe löschen wollen, geben Sie im Bestätigungsfeld den Namen der Benutzergruppe ein und wählen Löschen. Wenn Sie mehrere Benutzergruppen löschen möchten, geben Sie die Anzahl der zu löschenden Benutzergruppen ein, gefolgt von **`user groups`** und wählen Sie Löschen. Wenn Sie zum Beispiel drei Benutzergruppen löschen, geben Sie **`3 user groups`** ein.

Löschen einer IAM-Benutzergruppe (AWS CLI)

Sie können eine IAM-Benutzergruppe aus dem AWS CLI löschen.

So löschen Sie eine IAM-Gruppe (AWS CLI)

1. Entfernen Sie alle Benutzer aus der Gruppe.
   - aws iam get-group (um die Liste der Benutzer in der Benutzergruppe abzurufen) und aws iam remove-user-from-group (um einen Benutzer aus der Benutzergruppe zu entfernen)
2. Löschen Sie alle einbetteten Inlinerichtlinien in der Gruppe.
   - aws iam list-group-policies (um eine Liste der Inline-Richtlinien der Benutzergruppe abzurufen) und aws iam delete-group-policy (um die Inline-Richtlinien der Benutzergruppe zu löschen)
3. Trennen Sie alle der Gruppe angefügten verwalteten Richtlinien.
   - aws iam list-attached-group-policies (um eine Liste der verwalteten Richtlinien abzurufen, die der Benutzergruppe zugeordnet sind) und aws iam detach-group-policy (um eine verwaltete Richtlinie von der Benutzergruppe zu trennen)
4. Löschen Sie die Gruppe.
   - aws iam delete-group

Löschen einer IAM-Benutzergruppe (API)AWS

Sie können die AWS API verwenden, um eine IAM-Benutzergruppe zu löschen.

Um eine IAM-Benutzergruppe (AWS API) zu löschen

1. Entfernen Sie alle Benutzer aus der Gruppe.
   - GetGroup(um die Liste der Benutzer in der Benutzergruppe abzurufen) und RemoveUserFromGroup(um einen Benutzer aus der Benutzergruppe zu entfernen)
2. Löschen Sie alle einbetteten Inlinerichtlinien in der Gruppe.
   - ListGroupPolicies(um eine Liste der Inline-Richtlinien der Benutzergruppe zu erhalten) und DeleteGroupPolicy(um die Inline-Richtlinien der Benutzergruppe zu löschen)
3. Trennen Sie alle der Gruppe angefügten verwalteten Richtlinien.

- ListAttachedGroupPolicies(um eine Liste der verwalteten Richtlinien abzurufen, die der Benutzergruppe zugeordnet sind) und DetachGroupPolicy(um eine verwaltete Richtlinie von der Benutzergruppe zu trennen)

4. Löschen Sie die Gruppe.

   - DeleteGroup

IAM-Rollen

Eine IAM-Rolle ist eine IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. Eine IAM-Rolle ähnelt einem IAM-Benutzer insofern, als es sich um eine AWS Identität mit Berechtigungsrichtlinien handelt, die festlegen, was die Identität tun kann und was nicht. AWS Eine Rolle ist jedoch nicht einer einzigen Person zugeordnet, sondern kann von allen Personen angenommen werden, die diese Rolle benötigen. Einer Rolle sind außerdem keine standardmäßigen, langfristigen Anmeldeinformationen (Passwörter oder Zugriffsschlüssel) zugeordnet. Wenn Sie eine Rolle übernehmen, erhalten Sie stattdessen temporäre Anmeldeinformationen für Ihre Rollensitzung.

Sie können Rollen verwenden, um den Zugriff an Benutzer, Anwendungen oder Dienste zu delegieren, die normalerweise keinen Zugriff auf Ihre Ressourcen haben. AWS Möglicherweise möchten Sie Benutzern in Ihrem AWS Konto Zugriff auf Ressourcen gewähren, über die sie normalerweise nicht verfügen, oder Benutzern in einem Konto AWS-Konto Zugriff auf Ressourcen in einem anderen Konto gewähren. Oder Sie möchten einer mobilen App die Nutzung von AWS Ressourcen ermöglichen, aber keine AWS Schlüssel in die App einbetten (wo sie schwierig zu aktualisieren sein können und Benutzer sie möglicherweise extrahieren können). Manchmal möchten Sie Benutzern AWS Zugriff gewähren, für die bereits Identitäten außerhalb von definiert wurden AWS, z. B. in Ihrem Unternehmensverzeichnis. Oder Sie können auch Drittanbietern Zugriff auf Ihr Konto gewähren, sodass sie eine Prüfung Ihrer Ressourcen ausführen können.

In diesen Szenarien können Sie den Zugriff auf AWS Ressourcen mithilfe einer IAM-Rolle delegieren. Dieser Abschnitt bietet eine Einführung in Rollen und in die verschiedenen Verwendungsmöglichkeiten, wann und wie Sie unter den Methoden auswählen können und wie Sie Rollen erstellen, verwalten, annehmen und löschen bzw. zu diesen wechseln können.

> **Note**
>
> Wenn Sie Ihre Rollen zum ersten Mal erstellen AWS-Konto, werden standardmäßig keine Rollen erstellt. Wenn Sie Ihrem Konto Services hinzufügen, können sie zur Unterstützung ihrer Anwendungsfälle serviceverknüpfte Rollen hinzufügen.
> Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer AWS-Service verknüpft ist. 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.
> Bevor Sie die serviceverknüpften Rollen löschen können, müssen Sie zunächst die zugehörigen Ressourcen löschen. Dies schützt Ihre -Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.
> Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Themen

Rollenbegriffe und -konzepte

Im Folgenden finden Sie einige grundlegende Begriffe, die Ihnen bei den ersten Schritten mit Rollen helfen.

Rolle

Eine IAM-Identität, die Sie in Ihrem Konto erstellen können und die über bestimmte Berechtigungen verfügt. Es gibt gewisse Ähnlichkeiten zwischen einer IAM-Rolle und einem IAM-Benutzer. Sowohl Rollen als auch Benutzer sind AWS -Identitäten mit Berechtigungsrichtlinien, die bestimmen, was die Identität in AWS tun kann und was nicht. Eine Rolle ist jedoch nicht einer einzigen Person zugeordnet, sondern kann von allen Personen angenommen werden, die diese Rolle benötigen. Einer Rolle sind außerdem keine standardmäßigen, langfristigen Anmeldeinformationen (Passwörter oder Zugriffsschlüssel) zugeordnet. Wenn Sie eine Rolle übernehmen, erhalten Sie stattdessen temporäre Anmeldeinformationen für Ihre Rollensitzung.

Rollen können von folgenden Personen und Services verwendet werden:

- Ein IAM-Benutzer in derselben AWS-Konto Rolle
- Ein IAM-Benutzer in einer anderen Rolle AWS-Konto als der Rolle
- Ein Webservice, der AWS beispielsweise von Amazon Elastic Compute Cloud (Amazon EC2) angeboten wird
- Ein externer Benutzer, der von einem externen Identitätsanbieter(IdP)-Service authentifiziert wurde, der mit SAML 2.0 oder OpenID Connect kompatibel ist, oder ein kundenspezifischer Identity Broker.

AWS Servicerolle

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.

AWS Servicerolle für eine EC2-Instance

Eine besondere Art von Servicerolle, die eine Anwendung, die auf einer Amazon EC2-Instance ausgeführt wird, übernehmen kann, um Aktionen in Ihrem Konto auszuführen. Diese Rolle ist der EC2-Instance zugewiesen, wenn sie gestartet wird. Anwendungen, die auf dieser Instance ausgeführt werden, können temporäre Anmeldeinformationen abrufen und Aktionen ausführen, die die Rolle erlaubt. Weitere Informationen über die Verwendung einer Servicerolle für eine EC2-Instance finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden.

AWS Rolle, die mit einem Dienst verknüpft

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. 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.

> **Note**
>
> Wenn Sie einen Service bereits verwenden, wenn er mit der Unterstützung serviceverknüpfter Rollen beginnt, erhalten Sie möglicherweise eine E-Mail zur Benachrichtigung über die neue Rolle in Ihrem Konto. In diesem Fall erstellt der Service die serviceverknüpfte Rolle automatisch in Ihrem Konto. Sie müssen keine Aktion zum Unterstützen dieser Rolle durchführen und Sie sollten sie nicht manuell löschen. Weitere Informationen finden Sie unter In meinem AWS -Konto wird eine neue Rolle angezeigt.

Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen.

Verketten von Rollen

Bei der Rollenverkettung verwenden Sie eine Rolle, um über die OR-API eine zweite Rolle zu übernehmen. AWS CLI Zum Beispiel: `RoleA` verfügt über die Berechtigung `RoleB` anzunehmen. Sie können Benutzer1 die Übernahme ermöglichen, `RoleA` indem Sie dessen langfristige Benutzeranmeldedaten im AssumeRole API-Vorgang verwenden. Diese gibt die kurzfristigen Anmeldeinformationen von `RoleA` zurück. Für das Verketten von Rollen können Sie die kurzfristigen Anmeldeinformationen von `RoleA` verwenden, um User1 zu ermöglichen `RoleB` anzunehmen.

Wenn Sie eine Rolle übernehmen, können Sie ein Sitzungs-Tag übergeben und dieses als transitiv festlegen. Transitive Sitzungs-Tags werden an alle nachfolgenden Sitzungen in einer Rollenkette übergeben. Weitere Informationen zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

Durch die Rollenverkettung wird Ihre Rollensitzung AWS CLI oder Ihre AWS API-Rollensitzung auf maximal eine Stunde begrenzt. Wenn Sie den AssumeRoleAPI-Vorgang verwenden, um eine Rolle anzunehmen, können Sie die Dauer Ihrer Rollensitzung mit dem `DurationSeconds` Parameter angeben. Sie können einen Parameterwert von bis zu 43200 Sekunden (12 Stunden) angeben, abhängig von der Einstellung maximale Sitzungsdauer für Ihre Rolle. Wenn Sie eine Rolle durch Verkettung übernehmen und für den Parameter `DurationSeconds` einen Wert größer als eine Stunde angeben, schlägt die Operation jedoch fehl.

AWS behandelt die Verwendung von Rollen zur Erteilung von Berechtigungen für Anwendungen, die auf EC2-Instances ausgeführt werden, nicht als Rollenverkettung.

Delegierung

Die Erteilung von Berechtigungen für jemanden, um den Zugriff auf Ressourcen zu gewähren, die Sie kontrollieren. Die Delegierung beinhaltet die Einrichtung einer Vertrauensstellung zwischen zwei Konten. Das erste Konto ist Eigentümer der Ressource (das vertrauensvolle Konto). Das zweite Konto enthält die Benutzer, die auf die Ressource zugreifen müssen (das vertrauenswürdige Konto). Die vertrauenswürdigen und vertrauensvollen Konten können Folgende sein:

- Das gleiche Konto.
- Separate Konten, die beide von Ihrer Organisation kontrolliert werden.
- Zwei Konten im Besitz von unterschiedlichen Organisationen.

Zum Delegieren einer Berechtigung für den Zugriff auf eine Ressource erstellen Sie eine IAM-Rolle im vertrauenden Konto, der zwei Richtlinien angefügt sind. Die Berechtigungsrichtlinie erteilt dem Benutzer der Rolle die erforderlichen Berechtigungen zum Ausführen der vorgesehenen Aufgaben auf der Ressource. Die Vertrauensrichtlinie gibt an, welche vertrauenswürdigen Kontomitglieder die Rolle übernehmen dürfen.

Wenn Sie eine Vertrauensrichtlinie erstellen, können Sie keinen Platzhalter (*) als Teil eines ARN im Prinzipalelement angeben. Die Vertrauensrichtlinie ist an die Rolle im vertrauenden Konto angefügt und eine Hälfte der Berechtigungen. Die andere Hälfte ist eine Berechtigungsrichtlinie, die dem Benutzer im vertrauenswürdigen Konto angefügt ist und zulässt, dass ein Benutzer zur Rolle wechselt bzw. diese übernimmt. Ein Benutzer, der eine Rolle übernimmt, gibt vorübergehend seine eigenen Berechtigungen auf und übernimmt stattdessen die Berechtigungen der Rolle. Wenn der Benutzer die Rolle beendet oder nicht mehr verwendet, werden die ursprünglichen Benutzerberechtigungen wiederhergestellt. Ein zusätzlicher Parameter namens external ID kann helfen, die sichere Verwendung von Rollen zwischen Konten sicherzustellen, die nicht von der gleichen Organisation kontrolliert werden.

Verbund

Die Schaffung einer Vertrauensbeziehung zwischen einem externen Identitätsanbieter und. AWS Benutzer können sich bei einem OIDC-Anbieter anmelden, z. B. Login with Amazon, Facebook, Google oder einem beliebigen IdP, der mit OpenID Connect (OIDC) kompatibel ist. Benutzer können sich auch bei einem Unternehmens-Identitätssystem anmelden, das kompatibel mit Security Assertion Markup Language (SAML) 2.0, wie Microsoft Active Directory Federation Services, ist. Wenn Sie OIDC und SAML 2.0 verwenden, um eine Vertrauensbeziehung zwischen diesen externen Identitätsanbietern und zu konfigurieren AWS, wird dem Benutzer eine IAM-Rolle zugewiesen. Der Benutzer erhält außerdem temporäre Anmeldeinformationen, mit denen er auf Ihre Ressourcen zugreifen kann. AWS

Verbundbenutzer

Anstatt einen IAM-Benutzer zu erstellen, können Sie vorhandene Identitäten aus AWS Directory Service Ihrem Unternehmensbenutzerverzeichnis oder einem OIDC-Anbieter verwenden. Diese werden als Verbundbenutzer bezeichnet. AWS weist einem Verbundbenutzer eine Rolle zu, wenn der Zugriff über einen Identitätsanbieter angefordert wird. Weitere Informationen zu Verbundbenutzern finden Sie unter Verbundbenutzer und -rollen.

Vertrauensrichtlinie

Ein JSON-Richtliniendokument, in dem Sie die Auftraggeber definieren, denen Sie es anvertrauen, die Rolle zu übernehmen. Eine Rollenvertrauensrichtlinie ist eine erforderliche ressourcenbasierte Richtlinie die einer Rolle in IAM angefügt ist. Zu den Auftraggeber, die Sie in der Vertrauensrichtlinie angeben können, gehören Benutzer, Rollen, Konten und Services.

Berechtigungsrichtlinie

Ein Berechtigungsdokument im JSON-Format, in dem Sie definieren, welche Aktionen und Ressourcen die Rolle verwenden kann. Das Dokument wird entsprechend den Regeln der IAM-Richtliniensprache geschrieben.

Berechtigungsgrenze

Ein erweitertes Feature, in der Sie Richtlinien verwenden, um die maximalen Berechtigungen einzuschränken, die eine identitätsbasierte Richtlinie einer Rolle gewähren kann. Sie können eine Berechtigungsgrenze nicht auf eine servicegebundene Rolle anwenden. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten.

Auftraggeber

Eine Entität AWS , die Aktionen ausführen und auf Ressourcen zugreifen kann. Ein Principal kann ein Root-Benutzer des AWS-Kontos, ein IAM-Benutzer oder eine Rolle sein. Sie können Berechtigungen für den Zugriff auf eine Ressource auf eine von zwei Arten gewähren:

- Sie können eine Berechtigungsrichtlinie an einen Benutzer (direkt oder indirekt durch eine Gruppe) oder an eine Rolle anfügen.
- Für Services, die ressourcenbasierte Richtlinien unterstützen, können Sie den Auftraggeber im `Principal`-Element einer Richtlinie, die an die Ressource angefügt ist, identifizieren.

Wenn Sie auf einen AWS-Konto als Prinzipal verweisen, bedeutet dies in der Regel jeden Prinzipal, der in diesem Konto definiert ist.

> **Note**
>
> Sie können keinen Platzhalter (*) verwenden, um einen Teil eines Prinzipalnamens oder eines ARNs in der Vertrauensrichtlinie einer Rolle zu ersetzen. Details hierzu finden Sie unter AWS JSON-Richtlinienelemente: Principal.

Rolle für kontoübergreifenden Zugriff

Eine Rolle gewährt Ressourcen in einem Konto Zugriff auf einen vertrauenswürdigen Auftraggeber in einem anderen Konto. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Einige AWS -Services ermöglichen Ihnen, eine Richtlinie direkt an eine Ressource anzufügen (anstatt eine Rolle als Proxy zu verwenden). Diese werden als ressourcenbasierte Richtlinien bezeichnet und Sie können sie verwenden, um anderen Benutzern AWS-Konto Zugriff auf die Ressource zu gewähren. Einige dieser Ressourcen umfassen Amazon Simple Storage Service (S3) Buckets, S3 Glacier Tresore, Amazon Simple Notification Service (SNS) -Themen und Amazon Simple Queue Service (SQS) Warteschlangen. Weitere Informationen darüber, welche Services ressourcenbasierte Richtlinien unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Weitere Informationen zu ressourcenbasierten Richtlinien finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.

Gängige Szenarien für Rollen: Benutzer, Anwendungen und Services

Wie bei den meisten AWS Funktionen haben Sie generell zwei Möglichkeiten, eine Rolle zu verwenden: interaktiv in der IAM-Konsole oder programmgesteuert mit den AWS CLI Tools für Windows PowerShell oder der API.

- IAM-Benutzer in Ihrem Konto können mithilfe der IAM-Konsole eine Rolle wechseln, um vorübergehend die Berechtigungen der Rolle in der Konsole zu verwenden. Die Benutzer verlieren ihre ursprünglichen Berechtigungen und übernehmen die Berechtigungen der zugewiesenen Rolle. Wenn der Benutzer die Rolle verlässt, werden die ursprünglichen Berechtigungen wiederhergestellt.
- Eine Anwendung oder ein Service, der von AWS (wie Amazon EC2) angeboten wird, kann eine Rolle übernehmen, indem sie temporäre Sicherheitsanmeldedaten für eine Rolle anfordert, an die programmatische Anfragen gestellt werden können. AWS Wenn Sie eine Rolle auf diese Weise verwenden, müssen Sie keine langfristigen Anmeldeinformationen für jede Entität teilen oder pflegen (indem Sie beispielsweise einen IAM-Benutzer erstellen), die Zugriff auf eine Ressource benötigt.

> **Note**
>
> In diesem Handbuch stellen die Ausdrücke zu einer Rolle wechseln und eine Rolle übernehmen Synonyme dar.

Die einfachste Möglichkeit zur Verwendung von Rollen besteht darin, Ihren IAM-Benutzern Berechtigungen zum Wechseln der Rollen zu gewähren, die Sie in Ihrem oder einem anderen AWS-Konto erstellen. Sie können Rollen auf einfache Weise mit der IAM-Konsole wechseln, um Berechtigungen zu verwenden, über die die Rollen normalerweise nicht verfügen sollen, und dann die Rolle verlassen, um diese Berechtigungen zu entziehen. So können Sie den versehentlichen Zugriff auf oder die Änderung von sensible Ressourcen vermeiden.

Für komplexere Verwendungen von Rollen, z. B. Gewähren des Zugriffs auf externe Anwendungen und Services, rufen Sie das `AssumeRole`-API auf. Dieser API-Aufruf gibt einen Satz temporärer Anmeldeinformationen zurück, die die Anwendung in nachfolgenden API-Aufrufen verwenden kann. Aktionen, die mit den temporären Anmeldeinformationen aufgerufen werden, verfügen nur über die von der zugewiesenen Rolle gewährten Berechtigungen. Eine Anwendung muss die Rolle nicht wie der Benutzer über die Konsole "verlassen". Die Anwendung verwendet einfach nicht

mehr die temporären Anmeldeinformationen und führt die Aufrufe nunmehr mit den ursprünglichen Anmeldeinformationen durch.

Verbundbenutzer melden sich mit den Anmeldeinformationen eines Identitätsanbieters (IdP) an. AWS stellt dann dem vertrauenswürdigen IdP temporäre Anmeldeinformationen zur Verfügung, die an den Benutzer weitergegeben werden, damit er sie in nachfolgende AWS Ressourcenanfragen einbezieht. Diese Anmeldeinformationen enthalten die Berechtigungen für die zugewiesene Rolle.

Dieser Abschnitt enthält eine Übersicht über die folgenden Szenarien:

- Gewähren Sie einem IAM-Benutzer in einem Konto AWS-Konto , das Sie besitzen, Zugriff auf Ressourcen in einem anderen Konto, das Sie besitzen
- Bereitstellen des Zugriffs auf nicht- AWS -Workloads
- Gewähren Sie den IAM-Benutzern in AWS-Konten von Dritten Zugriff
- Ermöglichen Sie den Zugriff auf Dienste, die von AWS to AWS resources angeboten werden
- Gewähren Sie den Zugriff für extern authentifizierte Benutzer (Identitätsverbund)

Bereitstellen des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen

Sie können Ihren IAM-Benutzern die Erlaubnis erteilen, zu Rollen innerhalb Ihrer eigenen AWS-Konto oder zu Rollen zu wechseln, die in anderen Rollen definiert sind AWS-Konten , die Ihnen gehören.

> **Note**
>
> Wenn Sie den Zugriff auf ein Konto gewähren möchten, das Sie nicht besitzen oder kontrollieren, finden Sie weitere Informationen unter Bereitstellung des Zugriffs auf AWS-Konten Eigentum Dritter im weiteren Verlauf in diesem Thema.

Angenommen, Sie haben Amazon EC2-Instances, die für die Organisation sehr wichtig sind. Statt Benutzern die Berechtigung zu erteilen, solche Instances zu beenden, können Sie eine Rolle mit diesen Berechtigungen erstellen. Erlauben Sie dann den Administratoren, zu jener Rolle zu wechseln, wenn sie eine Instance beenden müssen. Dadurch werden den Instances folgende Schutzebenen hinzugefügt:

- Sie müssen Ihren Benutzern explizit die Berechtigung zur Übernahme der Rolle erteilen.

- Ihre Benutzer müssen aktiv zu der Rolle wechseln, indem sie die AWS Management Console oder AWS API verwenden, AWS CLI oder die Rolle übernehmen.
- Sie können die Multi-Factor Authentication (MFA) zur Rolle hinzufügen, sodass nur die Benutzer, die sich mit einem MFA-Gerät anmelden, die Rolle übernehmen können. Weitere Informationen dazu, wie Sie eine Rolle konfigurieren, sodass von den Benutzern, die die Rolle übernehmen müssen, zunächst eine Multi-Factor Authentication (MFA) verlangt wird, finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs.

Wir empfehlen diese Herangehensweise, um das das Prinzip der geringsten Zugriffsrechte durchzusetzen. Das bedeutet, dass erweiterte Berechtigungen nur in Zeiten gewährt werden, in denen diese zur Durchführung bestimmter Aufgaben benötigt werden. Mit den Rollen können Sie versehentliche Änderungen an sensiblen Umgebungen verhindern, vor allem in Verbindung mit der -Überwachung, um sicherzustellen, dass Rollen nur bei Bedarf verwendet werden.

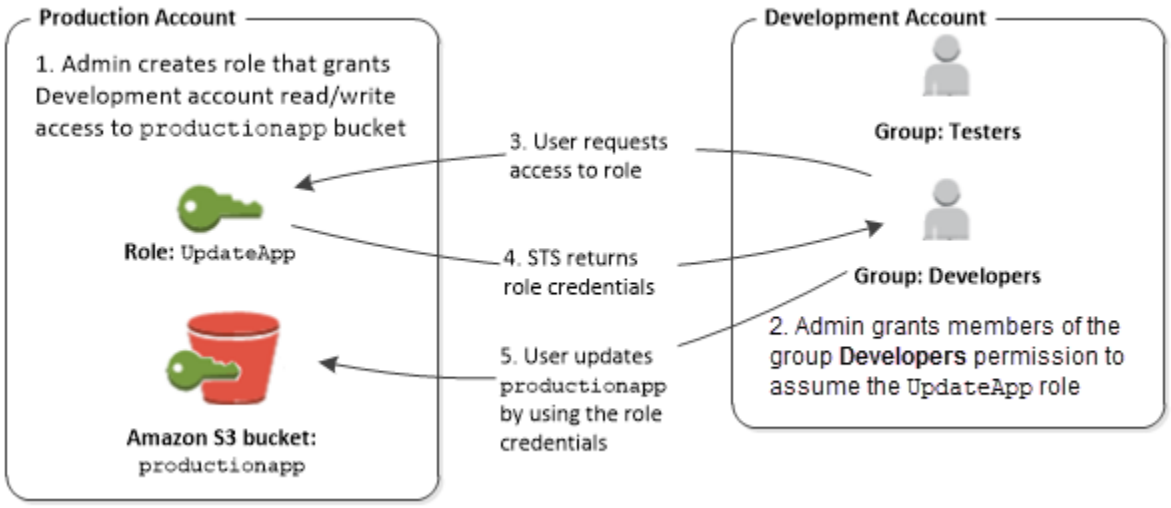
Wenn Sie eine Rolle für diesen Zweck erstellen, geben Sie in der Vertrauensrichtlinie der Rolle im `Principal`-Element die IDs der Konten an, für deren Benutzer Sie Zugriffsberechtigungen erteilen müssen. Anschließend können Sie bestimmten Benutzern in den anderen Konten Berechtigungen erteilen, um zu dieser Rolle zu wechseln. Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

Ein Benutzer in einem Konto kann zu einer Rolle in demselben oder einem anderen Konto wechseln. Während die Rolle verwendet wird, können die Benutzer nur die Aktionen durchführen und auf die Ressourcen zugreifen, die von der Rolle zugelassen sind. Die ursprünglichen Berechtigungen der Benutzer sind ausgesetzt. Wenn der Benutzer die Rolle verlässt, werden die ursprünglichen Benutzerberechtigungen wiederhergestellt.

Beispielszenario mit getrennten Entwicklungs- und Produktionskonten

Stellen Sie sich vor, Ihr Unternehmen verfügt über mehrere AWS-Konten , um eine Entwicklungsumgebung von einer Produktionsumgebung zu isolieren. Benutzer im Entwicklungskonto müssen gelegentlich auf Ressourcen im Produktionskonto zugreifen. Sie benötigen beispielsweise kontoübergreifenden Zugriff, wenn Sie ein Update aus der Entwicklungsumgebung in die Produktionsumgebung übernehmen wollen. Auch wenn Sie verschiedene Identitäten (und Passwörter) für die Benutzer, die in beiden Konten arbeiten, erstellen könnten, erschwert das Verwalten von Anmeldeinformationen für mehrere Konten die Identitätsverwaltung. In der folgenden Abbildung werden alle Benutzer im Development-Konto verwaltet. Einige Entwickler benötigen jedoch eingeschränkten Zugriff auf das Production-Konto.

Das Development-Konto verfügt über zwei Gruppen: Tester und Entwickler und jede Gruppe hat ihre eigene Richtlinie.



1. Im Produktionskonto verwendet ein Administrator IAM, um die `UpdateApp`-Rolle in diesem Konto zu erstellen. Der Administrator definiert in der Rolle eine Vertrauensrichtlinie, in der das Development-Konto als `Principal` angegeben wird, was bedeutet, dass autorisierte Benutzer aus dem Development-Konto die Rolle `UpdateApp` verwenden dürfen. Der Administrator definiert auch eine Berechtigungsrichtlinie für die Rolle, die festlegt, welche Rollenbenutzer Lese- und Schreibberechtigungen für den Amazon S3`productionapp`-Bucket namens erhalten.

   Der Administrator gibt die entsprechenden Informationen dann an Benutzer weiter, die die Rolle übernehmen müssen. Diese Informationen sind die Kontonummer und der Name der Rolle (für AWS Konsolenbenutzer) oder der Amazon-Ressourcenname (ARN) (für AWS CLI unseren AWS API-Zugriff). Der Rollen-ARN könnte wie `arn:aws:iam::123456789012:role/UpdateApp` aussehen, wobei die Rolle `UpdateApp` benannt ist und die Rolle unter der Kontonummer 123456789012 erstellt wurde.

   > **Note**
   >
   > Der Administrator kann optional die Rolle konfigurieren, sodass von den Benutzern, die die Rolle übernehmen müssen, zunächst eine Multi-Factor Authentication (MFA) verlangt wird. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs.

2. Der Administrator gewährt den Mitgliedern der Entwicklergruppe im Entwicklungskonto die Berechtigungen, um zur betreffenden Rolle wechseln zu können. Dies erfolgt, indem der

Entwicklergruppe die Erlaubnis erteilt wird, die AWS Security Token Service (AWS STS) `AssumeRole` -API für die `UpdateApp` Rolle aufzurufen. Alle IAM-Benutzer der Entwickler-Gruppe im Development-Konto können nun zur Rolle `UpdateApp` im Production-Konto wechseln. Andere Benutzer, die sich nicht in der Entwickler-Gruppe befinden, haben keine Berechtigung, um zur Rolle zu wechseln, und sind folglich nicht in der Lage, auf den S3-Bucket im Production-Konto zuzugreifen.

3. Der Benutzer fordert den Wechsel zur Rolle an:
   - AWS Konsole: Der Benutzer wählt den Kontonamen in der Navigationsleiste und wählt „Rolle wechseln“. Der Benutzer gibt die Konto-ID (oder Alias) und den Namen der Rolle an. Alternativ kann der Benutzer auf einen Link in der vom Administrator gesendeten E-Mail klicken. Der Link leitet den Benutzer zur Seite Switch Role (Rolle wechseln) weiter, in der die Details bereits ausgefüllt sind.
   - AWS API/AWS CLI: Ein Benutzer in der Gruppe Entwickler des Entwicklungskontos ruft die `AssumeRole` Funktion auf, um Anmeldeinformationen für die `UpdateApp` Rolle abzurufen. Der Benutzer übergibt den ARN der Rolle `UpdateApp` als Teil des Aufrufs. Eine vom Benutzer in der Tester-Gruppe gestellte gleiche Anforderung schlägt fehl, weil die Tester keine Berechtigung zum Aufruf von `AssumeRole` für den ARN der Rolle `UpdateApp` haben.
4. AWS STS gibt temporäre Anmeldeinformationen zurück:
   - AWS Konsole: AWS STS überprüft die Anfrage anhand der Vertrauensrichtlinie der Rolle, um sicherzustellen, dass die Anfrage von einer vertrauenswürdigen Entität stammt (was es ist: das Entwicklungskonto). AWS STS Gibt nach der Überprüfung temporäre Sicherheitsanmeldedaten an die AWS Konsole zurück.
   - API/CLI: AWS STS überprüft die Anfrage anhand der Vertrauensrichtlinie der Rolle, um sicherzustellen, dass die Anfrage von einer vertrauenswürdigen Entität stammt (was es ist: das Entwicklungskonto). AWS STS Gibt nach der Überprüfung temporäre Sicherheitsanmeldedaten an die Anwendung zurück.
5. Die temporären Anmeldeinformationen ermöglichen den Zugriff auf die AWS Ressource:
   - AWS Konsole: Die AWS Konsole verwendet die temporären Anmeldeinformationen im Namen des Benutzers für alle nachfolgenden Konsolenaktionen, in diesem Fall zum Lesen und Schreiben in den `productionapp` Bucket. Die Konsole kann auf keine anderen Ressourcen im Production-Konto zurückgreifen. Wenn der Benutzer die Rolle verlässt, erhält er seine ursprünglichen Berechtigungen, über die er vor dem Wechseln der Rolle verfügt hat, zurück.
   - API/CLI: Die Anwendung verwendet die temporären Anmeldeinformationen, um den Bucket `productionapp` zu aktualisieren. Mit den temporären Sicherheitsanmeldeinformationen kann die Anwendung nur Lese- und Schreibvorgänge im Bucket `productionapp` ausführen

und auf keine anderen Ressourcen im Production-Konto zugreifen. Die Anwendung muss die Rolle nicht verlassen, sondern unterlässt stattdessen die Verwendung der temporären Anmeldeinformationen und verwendet in den nachfolgenden API-Aufrufen die ursprünglichen Anmeldeinformationen.

Weitere Informationen

Weitere Informationen finden Sie hier:

- Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen

Zugang für nicht- AWS -Workloads gewähren

Eine IAM-Rolle ist ein Objekt in AWS Identity and Access Management (IAM), dem Berechtigungen zugewiesen wurden. Wenn Sie diese Rolle mit einer IAM-Identität oder einer Identität von außerhalb übernehmen AWS, erhalten Sie temporäre Sicherheitsanmeldedaten für Ihre Rollensitzung. Möglicherweise laufen in Ihrem Rechenzentrum oder in einer anderen Infrastruktur Workloads, für AWS die der Zugriff auf Ihre AWS Ressourcen nicht erforderlich ist. Anstatt langfristige Zugriffsschlüssel zu erstellen, zu verteilen und zu verwalten, können Sie Roles Anywhere (IAM AWS Identity and Access Management Roles Anywhere) verwenden, um Ihre Nicht-Workloads zu authentifizieren. AWS IAM Roles Anywhere verwendet X.509-Zertifikate Ihrer Zertifizierungsstelle (CA), um Identitäten zu authentifizieren und den Zugriff auf sichere Weise AWS-Services mit den temporären Anmeldeinformationen zu ermöglichen, die von einer IAM-Rolle bereitgestellt werden.

Um IAM Roles Anywhere zu verwenden, richten Sie eine CA mithilfe von AWS Private Certificate Authority ein oder verwenden eine CA aus Ihrer eigenen PKI-Infrastruktur. Nachdem Sie eine CA eingerichtet haben, erstellen Sie in IAM Roles Anywhere ein Objekt, das als Vertrauensanker bezeichnet wird, um das Vertrauen zwischen IAM Roles Anywhere und Ihrer CA für die Authentifizierung herzustellen. Anschließend können Sie Ihre vorhandenen IAM-Rollen konfigurieren oder neue Rollen erstellen, die dem IAM-Roles-Anywhere-Dienst vertrauen. Wenn sich Ihre AWS Nicht-Workloads über den Trust Anchor bei IAM Roles Anywhere authentifizieren, können sie temporäre Anmeldeinformationen für Ihre IAM-Rollen abrufen, um auf Ihre Ressourcen zuzugreifen. AWS

Weitere Informationen zum Konfigurieren von IAM Roles Anywhere finden Sie unter Was ist AWS Identity and Access Management Roles Anywhere im IAM-Roles-Anywhere-Benutzerhandbuch.

Bereitstellung des Zugriffs auf AWS-Konten Eigentum Dritter

Wenn Dritte Zugriff auf die AWS Ressourcen Ihrer Organisation benötigen, können Sie Rollen verwenden, um den Zugriff auf diese Ressourcen zu delegieren. Zum Beispiel könnte ein externer Benutzer einen Service zur Verwaltung Ihrer AWS -Ressourcen anbieten. Mit IAM-Rollen können Sie diesen Dritten Zugriff auf Ihre AWS Ressourcen gewähren, ohne Ihre AWS Sicherheitsanmeldedaten weitergeben zu müssen. Stattdessen kann der Drittanbieter auf Ihre AWS Ressourcen zugreifen, indem er eine Rolle übernimmt, die Sie in Ihrem AWS-Konto erstellen. Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

Externe Benutzer müssen Ihnen die folgenden Informationen bereitstellen, damit Sie eine von ihnen übernehmbare Rolle erstellen können:

- Die AWS-Konto ID des Drittanbieters. Sie geben deren AWS-Konto ID als Principal an, wenn Sie die Vertrauensrichtlinie für die Rolle definieren.
- Eine externe ID zur eindeutigen Zuordnung der Rolle. Die externe ID kann eine beliebige geheime Kennung sein, die Ihnen und dem Drittanbieter bekannt ist. Sie können beispielsweise eine Rechnungs-ID zwischen Ihnen und dem externen Benutzer verwenden. Benutzen Sie aber keine leicht zu erratenden Zeichenfolgen wie den Namen oder die Telefonnummer des externen Benutzers. Sie müssen diese ID festlegen, wenn Sie die Vertrauensrichtlinie für die Rolle definieren. Der externe Benutzer muss diese ID angeben, wenn er die Rolle übernimmt. Weitere Informationen über die externe ID finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.
- Die Berechtigungen, die der Drittanbieter benötigt, um mit Ihren AWS Ressourcen zu arbeiten. Sie müssen diese Berechtigungen festlegen, wenn Sie die Berechtigungsrichtlinie für die Rolle definieren. Diese Richtlinie bestimmt, welche Aktionen der externe Benutzer ausführen darf und auf welche Ressourcen er Zugriff hat.

Nachdem Sie die Rolle erstellt haben, müssen Sie dem externen Benutzer den Amazon Resource Name (ARN) der Rolle mitteilen. Dieser benötigt den ARN der Rolle, um sie übernehmen zu können.

> **Important**
>
> Wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren, können diese auf jede Ressource zugreifen, die Sie in der Richtlinie angeben. Die vom externen Benutzer genutzten

Ressourcen werden Ihnen in Rechnung gestellt. Stellen Sie sicher, dass Sie die Nutzung Ihrer Ressourcen sinnvoll einschränken.

So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren

Manchmal müssen Sie einem Dritten Zugriff auf Ihre AWS Ressourcen gewähren (Delegiertenzugriff). Ein wichtiger Faktor hierbei ist die externe ID, eine optionale Information, mit der Sie in Vertrauensrichtlinien von IAM-Rollen festlegen können, wer die Rolle übernehmen kann.

⚠ Important

AWS behandelt die externe ID nicht als geheim. Nachdem Sie ein Geheimnis wie ein Zugriffsschlüsselpaar oder ein Passwort erstellt haben AWS, können Sie es nicht erneut anzeigen. Die externe ID für eine Rolle kann von jedem Benutzer mit der Berechtigung zum Anzeigen der Rolle angezeigt werden.

In einer Umgebung mit mehreren Mandanten, in der Sie mehrere Kunden mit unterschiedlichen AWS Konten unterstützen, empfehlen wir, jeweils eine externe ID zu verwenden. AWS-Konto Diese ID sollte eine zufällige Zeichenfolge sein, die von der Drittpartei generiert wird.

Um zu verlangen, dass der Drittanbieter beim Übernehmen einer Rolle eine externe ID bereitstellt, aktualisieren Sie die Vertrauensrichtlinie der Rolle mit der externen ID Ihrer Wahl.

Um eine externe ID anzugeben, wenn Sie eine Rolle übernehmen, verwenden Sie die AWS API AWS CLI oder, um diese Rolle anzunehmen. Weitere Informationen finden Sie unter AssumeRoleSTS-API-Vorgang oder STS-Befehlsübernahme-CLI-Vorgang.

Nehmen wir zum Beispiel an, Sie beschließen, ein Drittanbieter namens Example Corp mit der Überwachung Ihrer Kosten AWS-Konto und der Kostenoptimierung zu beauftragen. Um Ihre täglichen Ausgaben verfolgen zu können, muss Example Corp auf Ihre AWS Ressourcen zugreifen. Example Corp überwacht zudem viele weitere AWS -Konten anderer Kunden.

Gewähren Sie Example Corp keinen Zugriff auf einen IAM-Benutzer und dessen langfristige Anmeldeinformationen in Ihrem AWS -Konto. Verwenden Sie stattdessen eine IAM-Rolle und deren temporäre Sicherheitsanmeldeinformationen. Eine IAM-Rolle bietet einen Mechanismus, mit dem Dritte auf Ihre AWS Ressourcen zugreifen können, ohne langfristige Anmeldeinformationen (z. B. einen IAM-Benutzerzugriffsschlüssel) weitergeben zu müssen.

Sie können eine IAM-Rolle verwenden, um eine vertrauenswürdige Beziehung zwischen Ihrem Konto AWS-Konto und dem Example Corp-Konto aufzubauen. Nachdem diese Beziehung hergestellt wurde, kann ein Mitglied des Example Corp-Kontos die AWS Security Token Service AssumeRoleAPI aufrufen, um temporäre Sicherheitsanmeldeinformationen zu erhalten. Die Mitglieder von Example Corp können die Anmeldeinformationen dann verwenden, um auf AWS Ressourcen in Ihrem Konto zuzugreifen.

> **Note**
>
> Weitere Informationen zu den AssumeRole und anderen AWS API-Vorgängen, die Sie aufrufen können, um temporäre Sicherheitsanmeldedaten zu erhalten, finden Sie unterAnfordern temporärer Sicherheitsanmeldeinformationen.

Nachfolgend ist dieses Szenario detaillierter aufgeschlüsselt:

1. Sie beauftragen Beispielunternehmen und diese erstellen eine eindeutige Kunden-ID für Sie. Sie stellen Ihnen diese eindeutige Kunden-ID und ihre AWS-Konto Nummer zur Verfügung. Sie benötigen diese Informationen, um im nächsten Schritt eine IAM-Rolle zu erstellen.

   > **Note**
   >
   > Example Corp kann einen beliebigen Zeichenkettenwert für die verwenden ExternalId, sofern dieser für jeden Kunden eindeutig ist. Es kann sich dabei um eine Kundenkontonummer oder eine zufällige Zeichenfolge handeln, solange nicht für zwei Kunden derselbe Wert verwendet wird. Diese Nummer muss nicht geheim gehalten werden. Example Corp muss den ExternalId Wert jedem Kunden zur Verfügung stellen. Wichtig dabei ist, dass diese ID von Unternehmen XY und nicht von den Kunden generiert wird, um sicherzustellen, dass jede externe ID einzigartig ist.

2. Sie melden sich an AWS und erstellen eine IAM-Rolle, die Example Corp Zugriff auf Ihre Ressourcen gewährt. Wie bei allen IAM-Rollen verfügt die Rolle über zwei Richtlinien, eine Berechtigungsrichtlinie und eine Vertrauensrichtlinie. Über die Vertrauensrichtlinie der Rolle wird festgelegt, wer die Rolle annehmen kann. In unserem Beispielszenario gibt die Richtlinie die AWS-Konto Anzahl von Example Corp als an`Principal`. Dies ermöglicht Identitäten dieses Kontos, die Rolle anzunehmen. Außerdem fügen Sie der Vertrauensrichtlinie ein `Condition`-Element hinzu. Mit diesem `Condition` wird der Kontextschlüssel `ExternalId` getestet, um sicherzustellen, dass er mit der eindeutigen Kunden-ID von Beispielunternehmen übereinstimmt.

```
    "Principal": {"AWS": "Example Corp's AWS-Konto ID"},
    "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. In der Berechtigungsrichtlinie der Rolle ist festgelegt, welche Berechtigungen mit der Rolle verliehen werden. Sie können zum Beispiel festlegen, dass die Rolle nur die Verwaltung Ihrer Amazon EC2- und Amazon RDS-Ressourcen erlaubt, nicht aber die Verwaltung Ihrer IAM-Benutzer oder -Gruppen. In unserem Beispielszenario können Sie Beispielunternehmen über die Berechtigungsrichtlinie schreibgeschützten Zugriff auf sämtliche Ressourcen in Ihrem Konto gewähren.
4. Nachdem Sie die Rolle erstellt haben, stellen Sie Beispielunternehmen den Amazon-Ressourcennamen (ARN) der Rolle bereit.
5. Wenn Example Corp auf Ihre AWS Ressourcen zugreifen muss, ruft jemand aus dem Unternehmen die AWS `sts:AssumeRole` API auf. Der Aufruf beinhaltet den ARN der Rolle, die übernommen werden soll, und den ExternalId Parameter, der ihrer Kunden-ID entspricht.

Wenn die Anfrage von jemandem stammt, der Example Corp's verwendet AWS-Konto, und wenn der Rollen-ARN und die externe ID korrekt sind, ist die Anfrage erfolgreich. Anschließend werden temporäre Sicherheitsanmeldedaten bereitgestellt, mit denen Example Corp auf die AWS Ressourcen zugreifen kann, die Ihre Rolle zulässt.

Anders ausgedrückt, wenn eine Rollenrichtlinie eine externe ID enthält, muss jemand, der die Rolle annehmen möchte, als Auftraggeber in der Rolle angegeben sein und muss die korrekte externe ID angeben.

Warum eine externe ID verwenden?

Abstrakt ausgedrückt wird über die externe ID festgelegt, unter welchen Umständen Benutzer, die diese Rolle annehmen, arbeiten können. Außerdem kann der Kontoinhaber die Umstände festlegen, unter denen die Rolle angenommen werden kann. Die primäre Funktion des externen Ausweises besteht darin, Das Confused-Deputy-Problem zu lösen und zu verhindern.

Wann sollte ich die externe ID verwenden?

Verwenden Sie in folgenden Situationen eine externe ID:

- Sie sind AWS-Konto Eigentümer und haben eine Rolle für einen Drittanbieter konfiguriert, der zusätzlich zu Ihrer Rolle AWS-Konten auf andere zugreift. Sie sollten den Dritten um eine externe

ID bitten, die er mit angeben muss, wenn er Ihre Rolle annimmt. Dann führen Sie eine Prüfung durch, um zu testen, ob diese externe ID in der Vertrauensrichtlinie Ihrer Rolle vorhanden ist. Auf diese Weise wird sichergestellt, dass die externe Partei Ihre Rolle nur annehmen kann, wenn sie in Ihrem Namen handelt.

- Sie sind wie Beispielunternehmen in unserem vorherigen Szenario in einer Position, in der Sie Rollen für verschiedene Kunden annehmen, Sie sollten jedem Kunden eine eindeutige externe ID zuweisen und Ihre Kunden anweisen, die externe ID in die Vertrauensrichtlinie ihrer Rolle aufzunehmen. Anschließend müssen Sie in allen Anforderungen, in der Sie Rollen annehmen möchten, die korrekte externe ID angeben.

  Sie verfügen wahrscheinlich bereits über eine eindeutige ID für jeden Kunden, die Sie als externe ID verwenden können. Bei der externen ID handelt es sich nicht um einen Wert, den Sie explizit erstellen oder separat nur für diesen Zweck nachverfolgen müssen.

  Geben Sie die externe ID in allen `AssumeRole`-API-Aufrufen an. Wenn Sie von einem Kunden einen Rollen-ARN erhalten, testen Sie darüber hinaus mit und ohne die korrekte externe ID, ob Sie die Rolle annehmen können. Wenn Sie die Rolle ohne die korrekte externe ID annehmen können, speichern Sie den Rollen-ARN des Kunden nicht in Ihrem System. Warten Sie, bis der Kunde die Vertrauensrichtlinie der Rolle aktualisiert hat und eine Überprüfung der externen ID vornimmt. So helfen Sie Ihren Kunden, sich korrekt zu verhalten und sich selbst und Ihr Unternehmen vor dem Problem des verwirrten Stellvertreters zu schützen.

Gewähren von Zugriff auf einen AWS -Service

Bei vielen AWS Diensten müssen Sie mithilfe von Rollen steuern, auf welche Dienste zugreifen können. Eine Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen durchzuführen, wird als Servicerolle bezeichnet. Wenn eine Rolle für einen Service einem speziellen Zweck dient, kann sie als Servicerolle für EC2-Instances oder als mit dem Service verknüpfte Rolle kategorisiert werden. In der AWS -Dokumentation für die einzelnen Services finden Sie Informationen dazu, ob der jeweilige Service Rollen verwendet und wie dem Service eine Rolle zur Nutzung zugewiesen wird.

Einzelheiten zum Erstellen einer Rolle zur Delegierung des Zugriffs auf einen von angebotenen Dienst finden Sie AWS unterErstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS -Service.

Das Confused-Deputy-Problem

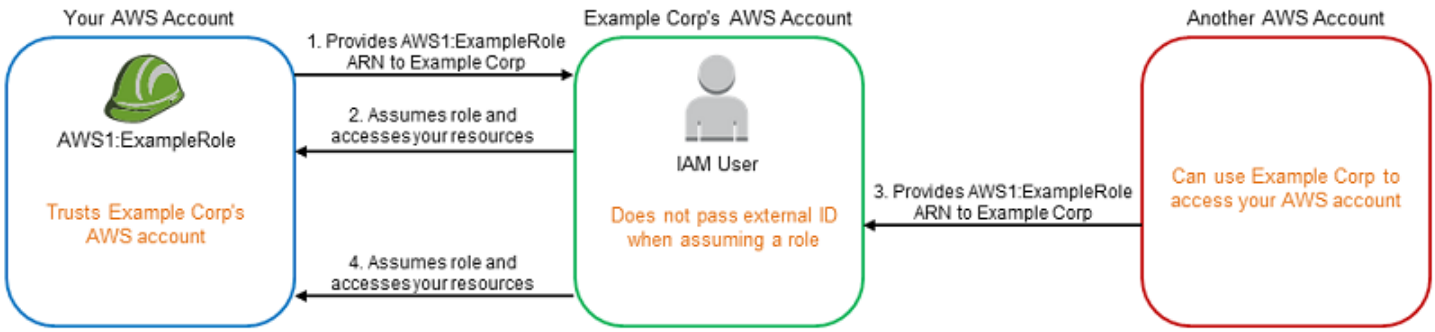
Das Problem des verwirrten Stellvertreters ist ein Sicherheitsproblem, bei dem eine Entität, die keine Berechtigung zur Durchführung einer Aktion hat, eine privilegiertere Entität zur Durchführung der Aktion zwingen kann. Um dies zu verhindern, AWS stellt Tools bereit, die Ihnen helfen, Ihr Konto zu schützen, wenn Sie Dritten (bekannt als kontoübergreifender Zugriff) oder anderen AWS Diensten (bekannt als dienstübergreifender Zugriff) Zugriff auf Ressourcen in Ihrem Konto gewähren.

Manchmal müssen Sie möglicherweise einer dritten Partei Zugriff auf Ihre AWS Ressourcen gewähren (Delegiertenzugriff). Nehmen wir zum Beispiel an, Sie beschließen, ein Drittanbieter namens Example Corp damit zu beauftragen, Ihre Kosten zu überwachen AWS-Konto und Sie bei der Kostenoptimierung zu unterstützen. Um Ihre täglichen Ausgaben verfolgen zu können, muss Example Corp auf Ihre AWS Ressourcen zugreifen. Example Corp überwacht auch viele andere AWS-Konten für andere Kunden. Sie können eine IAM-Rolle verwenden, um eine vertrauenswürdige Beziehung zwischen Ihrem Konto AWS-Konto und dem Example Corp-Konto aufzubauen. Ein wichtiger Faktor hierbei ist die externe ID, eine optionale Information, mit der Sie in Vertrauensrichtlinien von IAM-Rollen festlegen können, wer die Rolle übernehmen kann. Die Hauptfunktion des externen Ausweises besteht darin, das Problem des verwirrten Stellvertreters zu lösen und zu verhindern.

In AWS kann ein dienstübergreifendes Identitätswechsels zu einem Problem mit dem verwirrten Stellvertreter führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der Anruf-Dienst kann so manipuliert werden, dass er seine Berechtigungen verwendet, um auf die Ressourcen eines anderen Kunden zu reagieren, auf die er sonst nicht zugreifen dürfte.

Vermeidung des Problems des verwirrten Stellvertreters (kontoübergreifend)

In der nachfolgenden Abbildung wird das verwirrte Stellvertreter-Problem dargestellt.



In diesem Szenario wird von Folgendem ausgegangen:

- AWS 1 ist dein. AWS-Konto
- AWS 1: ExampleRole ist eine Rolle in Ihrem Konto. Über die Vertrauensrichtlinie dieser Rolle wird das AWS -Konto von Example Corp angegeben und kann somit die Rolle übernehmen.

Es passiert Folgendes:

1. Wenn Sie beginnen, den Service von Example Corp zu nutzen, geben Sie den ARN AWS 1: ExampleRole an Example Corp.
2. Example Corp verwendet diesen Rollen-ARN, um temporäre Sicherheitsanmeldeinformationen für den Zugriff auf Ressourcen in Ihrem zu erhalten AWS-Konto. Sie machen Example Corp also zu Ihrem "Stellvertreter", der in Ihrem Namen handeln darf.
3. Ein anderer AWS Kunde beginnt ebenfalls, den Service von Example Corp zu nutzen, und dieser Kunde stellt auch den ARN AWS 1 zur Verfügung: ExampleRole für Example Corp zur Nutzung. Vermutlich hat der andere Kunde die AWS 1: gelernt oder erratenExampleRole, was kein Geheimnis ist.
4. Wenn der andere Kunde Example Corp bittet, auf AWS Ressourcen in seinem Konto zuzugreifen (was es vorgibt), verwendet Example Corp AWS 1:, ExampleRole um auf Ressourcen in Ihrem Konto zuzugreifen.

So kann der andere Kunde unautorisiert Zugriff auf Ihre Ressourcen erlangen. Da der Kunde Example Corp dahingehend manipuliert hat, unwissentlich auf Ihre Ressourcen zuzugreifen, ist Beispielunternehmen nun ein "verwirrter Stellvertreter".

Example Corp vermeidet dieses Problem zum Beispiel, indem sie verlangt, in der Vertrauensrichtlinie der Rolle eine `ExternalId`-Bedingungsprüfung aufzunehmen. Example Corp generiert zum Beispiel einen einzigartigen`ExternalId`-Wert für jeden Kunden und verwendet diesen Wert in seiner Anforderung, um die Rolle zu übernehmen. Der `ExternalId`-Wert muss unter den Kunden von Example Corp eindeutig sein und von Example Corp kontrolliert werden, nicht von seinen Kunden. Deshalb erhalten Sie diese ID auch von Example Corp und können sie nicht selbst erstellen. Dadurch wird verhindert, dass Example Corp ein verwirrter Stellvertreter ist und Zugriff auf die AWS Ressourcen eines anderen Kontos gewährt.

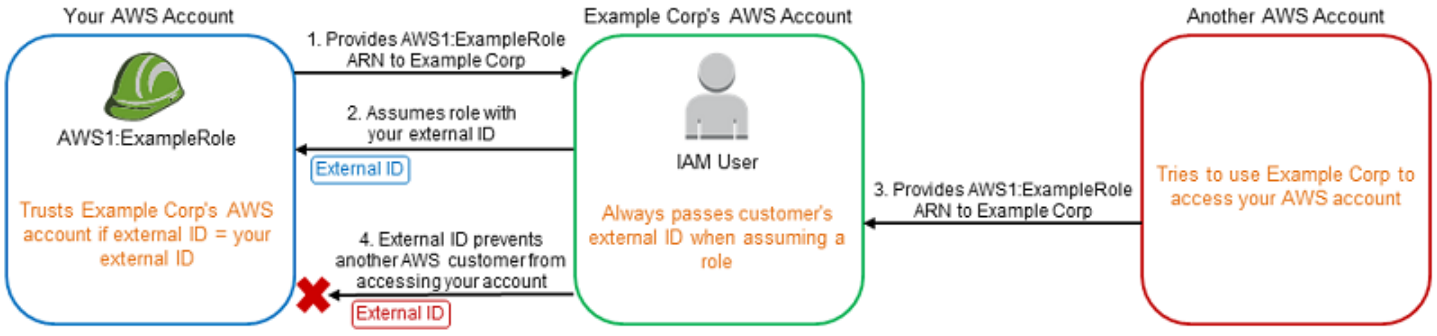
Stellen Sie sich in unserem Szenario vor, dass Ihre eindeutige ID beim Example Corp 12345 lautet und die ID des anderen Kunden 67890. Die IDs sind für dieses Szenario vereinfacht. Im Allgemeinen sind diese IDs GUIDs. Angenommen, diese IDs sind innerhalb des Kundenstamms von Example Corp eindeutig und es handelt sich um sinnvolle externe ID-Werte.

Example Corp teilt Ihnen den externen ID-Wert 12345 mit. Sie fügen nun ein `Condition`-Element zur Vertrauensrichtlinie der Rolle hinzu, dessen `sts:ExternalId`-Wert 12345 lauten muss. Beispiel:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
}
```

Das Condition-Element in dieser Richtlinie ermöglicht es Example Corp, die Rolle nur zu übernehmen, wenn der AssumeRole API-Aufruf den externen ID-Wert 12345 enthält. Example Corp stellt sicher, dass jedes Mal, wenn es eine Rolle im Namen eines Kunden übernimmt, immer den externen ID-Wert dieses Kunden in den AssumeRole Anruf mit einbezieht. Selbst wenn ein anderer Kunde Example Corp Ihren ARN zur Verfügung stellt, kann er nicht kontrollieren, welche externe ID Example Corp in seiner Anfrage an Example Corp angibt AWS. So wird verhindert, dass nicht autorisierte Kunden Zugriff auf Ihre Ressourcen erhalten.

Das folgende Diagramm verdeutlicht dieses Konzept.



1. Wenn Sie den Service von Example Corp in Anspruch nehmen, geben Sie wie bisher den ARN AWS 1: ExampleRole an Example Corp.

2. Wenn Example Corp diesen Rollen-ARN verwendet, um die Rolle AWS 1: anzunehmenExampleRole, nimmt Example Corp Ihre externe ID (12345) in den AssumeRole API-Aufruf auf. Die externe ID entspricht der Vertrauensrichtlinie der Rolle, sodass der AssumeRole API-Aufruf erfolgreich ist und Example Corp temporäre Sicherheitsanmeldedaten für den Zugriff auf Ressourcen in Ihrem erhält. AWS-Konto
3. Ein anderer AWS Kunde beginnt ebenfalls, den Service von Example Corp zu nutzen, und wie zuvor stellt dieser Kunde auch den ARN von AWS 1 zur Verfügung: ExampleRole für Example Corp zur Nutzung.
4. Aber dieses Mal, wenn Example Corp versucht, die Rolle AWS 1: anzunehmenExampleRole, gibt es die externe ID an, die dem anderen Kunden zugeordnet ist (67890). Der andere Kunde hat keinen Einfluss darauf. Example Corp gibt dies an, da die Anforderung zur Verwendung der Rolle von dem anderen Kunden stammte und 67890 somit die Umstände angibt, unter denen Example Corp handelt. Da Sie der Vertrauensrichtlinie AWS 1: eine Bedingung mit Ihrer eigenen externen ID (12345) hinzugefügt habenExampleRole, schlägt der AssumeRole API-Aufruf fehl. Der andere Kunde wird daran gehindert, unbefugten Zugriff auf Ressourcen in Ihrem Konto zu erhalten (gekennzeichnet durch das rote "X" in der Grafik).

Die externe ID verhindert, dass andere Kunden Example Corp manipuliert, und unwissentlich Zugriff auf Ihre Ressourcen gewährt.

Vermeidung des Problems des verwirrten Stellvertreters (dienstübergreifend)

Wir empfehlen die Verwendung der globalen Bedingungskontextschlüssel `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, oder `aws:SourceOrgPaths` in ressourcenbasierten Richtlinien, um die Berechtigungen zu beschränken, die ein Service für eine bestimmte Ressource hat. Wird verwendet`aws:SourceArn`, um nur eine Ressource mit dienstübergreifendem Zugriff zu verknüpfen. Wird verwendet`aws:SourceAccount`, damit jede Ressource in diesem Konto der dienstübergreifenden Nutzung zugeordnet werden kann. Wird verwendet`aws:SourceOrgID`, um zu ermöglichen, dass jede Ressource aus einem beliebigen Konto innerhalb einer Organisation mit der dienstübergreifenden Nutzung verknüpft wird. Wird verwendet`aws:SourceOrgPaths`, um jede Ressource aus Konten innerhalb eines AWS Organizations Pfads der dienstübergreifenden Nutzung zuzuordnen. Weitere Informationen zu Pfaden und deren Verwendung finden Sie unter Den AWS Organizations -Entitätspfad verstehen.

Am effektivsten schützen Sie sich vor dem Confused-Deputy-Problem, indem Sie den globalen Bedingungskontextschlüssel `aws:SourceArn` mit dem vollständigen ARN der Ressource in Ihren ressourcenbasierten Richtlinien verwenden. Wenn Sie den vollständigen ARN der Ressource

nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Bedingungskontextschlüssel `aws:SourceArn` mit Platzhaltern (*) für die unbekannten Teile des ARN. z. B. `arn:aws:servicename:*:123456789012:*`.

Wenn der `aws:SourceArn`-Wert die Konto-ID nicht enthält, z. B. einen Amazon-S3-Bucket-ARN, müssen Sie sowohl `aws:SourceAccount` als auch `aws:SourceArn` verwenden, um Berechtigungen einzuschränken.

Um sich vor dem Confused-Deputy-Problem im großen Maßstab zu schützen, verwenden Sie den globalen Bedingungskontextschlüssel `aws:SourceOrgID` oder `aws:SourceOrgPaths` mit der Organisations-ID oder dem Organisationspfad der Ressource in Ihren ressourcenbasierten Richtlinien. Richtlinien, die den Schlüssel `aws:SourceOrgID` oder `aws:SourceOrgPaths` enthalten, schließen automatisch die richtigen Konten ein und Sie müssen die Richtlinien nicht manuell aktualisieren, wenn Sie Konten in Ihrer Organisation hinzufügen, entfernen oder verschieben.

Bei non-service-linked Rollenvertrauensrichtlinien hat jeder Dienst in der Vertrauensrichtlinie die `iam:PassRole` Aktion durchgeführt, um zu überprüfen, ob sich die Rolle in demselben Konto befindet wie der aufrufende Dienst. Daher ist die Verwendung von `aws:SourceAccount`, `aws:SourceOrgID` oder `aws:SourceOrgPaths` mit diesen Vertrauensrichtlinien nicht erforderlich. Mithilfe von `aws:SourceArn` in einer Vertrauensrichtlinie können Sie Ressourcen angeben, für die eine Rolle übernommen werden kann, z. B. einen Lambda-Funktions-ARN. Einige AWS-Services verwenden `aws:SourceAccount` `aws:SourceArn` Vertrauensrichtlinien für neu erstellte Rollen, aber die Verwendung der Schlüssel ist für bestehende Rollen in Ihrem Konto nicht erforderlich.

> **Note**
>
> AWS-Services die in die AWS Key Management Service Verwendung von KMS-Schlüsselzuweisungen integriert sind, unterstützen die Schlüsselaws:SourceArn, `aws:SourceAccount`aws:SourceOrgID, oder die `aws:SourceOrgPaths` Bedingungsschlüssel nicht. Die Verwendung dieser Bedingungsschlüssel in einer KMS-Schlüsselrichtlinie führt zu unerwartetem Verhalten, wenn der Schlüssel auch AWS-Services über KMS-Schlüsselzuweisungen verwendet wird.

Dienstübergreifende verwirrte Stellvertreterprävention für AWS Security Token Service

Bei vielen AWS Diensten müssen Sie Rollen verwenden, damit der Dienst in Ihrem Namen auf die Ressourcen eines anderen Dienstes zugreifen kann. Eine Rolle, die ein Service übernimmt,

um Aktionen in Ihrem Namen durchzuführen, wird als Servicerolle bezeichnet. Eine Rolle erfordert zwei Richtlinien: eine Rollenvertrauensrichtlinie, die den Prinzipal angibt, der die Rolle übernehmen darf, und eine Berechtigungsrichtlinie, die angibt, was mit der Rolle gemacht werden kann. Eine Rollenvertrauensrichtlinie ist die einzige ressourcenbasierte Richtlinie in IAM. Andere AWS-Services haben ressourcenbasierte Richtlinien, wie z. B. eine Amazon S3 S3-Bucket-Richtlinie.

Wenn ein Dienst eine Rolle in Ihrem Namen übernimmt, muss der Service-Prinzipal die`sts:AssumeRole`-Aktion in der Rollenvertrauensrichtlinie ausführen dürfen. Wenn ein Service aufruft`sts:AssumeRole`, wird ein Satz temporärer Sicherheitsanmeldedaten AWS STS zurückgegeben, die der Service Principal verwendet, um auf die Ressourcen zuzugreifen, die gemäß der Berechtigungsrichtlinie der Rolle zulässig sind. Wenn ein Service eine Rolle in Ihrem Konto übernimmt, können Sie die globalen Bedingungskontextschlüssel `aws:SourceArn` `aws:SourceAccount`, `aws:SourceOrgID`, oder `aws:SourceOrgPaths` in Ihrer Rollenvertrauensrichtlinie übernehmen, um den Zugriff auf die Rolle nur auf Anffragen zu beschränken, die von erwarteten Ressourcen generiert werden.

In müssen Sie beispielsweise eine Rolle auswählen AWS Systems Manager Incident Manager, damit Incident Manager in Ihrem Namen ein Systems Manager Manager-Automatisierungsdokument ausführen kann. Das Automatisierungsdokument kann automatisierte Reaktionspläne für Vorfälle enthalten, die durch CloudWatch Alarme oder EventBridge Ereignisse ausgelöst werden.
Im folgenden Beispiel für eine Rollenvertrauensrichtlinie können Sie den `aws:SourceArn`-Bedingungsschlüssel verwenden, um den Zugriff auf die Servicerolle basierend auf dem ARN des Vorfalldatensatzes einzuschränken. Nur Vorfallsdatensätze, die aus der Reaktionsplan-Ressource `myresponseplan` erstellt werden, können diese Rolle verwenden.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "ssm-incidents.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ssm-incidents:*:111122223333:incident-
record/myresponseplan/*"
      }
    }
  }
```

```
}
```

> **Note**
>
> Nicht alle Serviceintegrationen mit AWS STS Support-`aws:SourceArn`, `aws:SourceAccountaws:SourceOrgID`, oder `aws:SourceOrgPaths` Bedingungsschlüsseln. Die Verwendung dieser Schlüssel in IAM-Vertrauensrichtlinien mit nicht unterstützten Integrationen kann zu unerwartetem Verhalten führen.

Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund)

Ihre Benutzer haben möglicherweise bereits Identitäten außerhalb von AWS, z. B. in Ihrem Unternehmensverzeichnis. Wenn diese Benutzer mit AWS Ressourcen arbeiten müssen (oder mit Anwendungen arbeiten müssen, die auf diese Ressourcen zugreifen), benötigen diese Benutzer auch AWS Sicherheitsanmeldeinformationen. Sie können mithilfe einer IAM-Rolle Berechtigungen für Benutzer festlegen, deren Identität in einem Verbund Ihres Unternehmens oder eines externen Identitätsanbieters (IdP) vereinigt ist.

> **Note**
>
> Als bewährte Sicherheitsmethode empfehlen wir, den Benutzerzugriff in IAM Identity Center mit einem Identitätsverbund zu verwalten, anstatt IAM-Benutzer zu erstellen. Informationen zu bestimmten Situationen, in denen ein IAM-Benutzer erforderlich ist, finden Sie unter Wann sollte ein IAM-Benutzer (anstelle einer Rolle) erstellt werden?.

Zusammenführung von Benutzern einer mobilen oder webbasierten Anwendung mit Amazon Cognito

Wenn Sie eine mobile oder webbasierte App erstellen, die auf AWS Ressourcen zugreift, benötigt die App Sicherheitsanmeldedaten, um programmatische Anfragen an sie stellen zu können. AWS Für die meisten mobilen Anwendungsszenarien empfehlen wir die Verwendung von Amazon Cognito. Sie können diesen Dienst mit dem AWS Mobile SDK for iOS und dem AWS Mobile SDK for Android und Fire OS verwenden, um eindeutige Identitäten für Benutzer zu erstellen und sie für einen sicheren Zugriff auf Ihre AWS Ressourcen zu authentifizieren. Amazon Cognito unterstützt die Identitätsanbieter, die im nächsten Abschnitt aufgeführt werden, sowie die vom Entwickler authentifizierten Identitäten und nicht authentifizierten Zugriff (Gast). Amazon Cognito stellt auch API-Operationen für die Synchronisierung von Benutzerdaten bereit, sodass diese erhalten bleiben, wenn

die Benutzer zwischen Geräten wechseln. Weitere Informationen finden Sie unter Verwenden von Amazon Cognito für mobile Apps.

Vereinigen von Benutzern in einem Verbund mit öffentlichen Identitätsdienstanbietern oder OpenID Connect

Verwenden Sie möglichst Amazon Cognito für mobile und webbasierte Anwendungsszenarien. Amazon Cognito erledigt den Großteil der behind-the-scenes Arbeit mit öffentlichen Identitätsanbieterdiensten für Sie. Es funktioniert mit den gleichen Drittanbieterservices und unterstützt auch anonyme Anmeldungen. Bei anspruchsvolleren Szenarien können Sie jedoch direkt mit einem Drittanbieterservice arbeiten, wie Login with Amazon, Facebook, Google oder einem mit OpenID Connect (OIDC) kompatiblen Identitätsanbieter. Weitere Informationen zur Verwendung des OIDC-Verbunds mit einem dieser Dienste finden Sie unter. OIDC-Föderation

Vereinigen von Benutzern in einem Verbund mit SAML 2.0

Wenn Ihre Organisation bereits ein Identity Provider-Softwarepaket verwendet, das SAML 2.0 (Security Assertion Markup Language 2.0) unterstützt, können Sie Vertrauen zwischen Ihrer Organisation als Identity Provider (IdP) und AWS als Service Provider aufbauen. Anschließend können Sie SAML verwenden, um Ihren Benutzern föderiertes Single Sign-On (SSO) AWS Management Console oder Verbundzugriff zum Aufrufen von API-Vorgängen bereitzustellen. AWS Wenn Ihr Unternehmen beispielsweise Microsoft Active Directory und Active Directory Federation Services verwendet, können Sie dies mit SAML 2 in einem Verbund vereinigen. Weitere Informationen zum Verbund von Benutzern mit SAML 2.0 finden Sie unter SAML 2.0-Verbund.

Vereinigen von Benutzern in einem Verbund durch Erstellen einer benutzerdefinierten Identity Broker-Anwendung

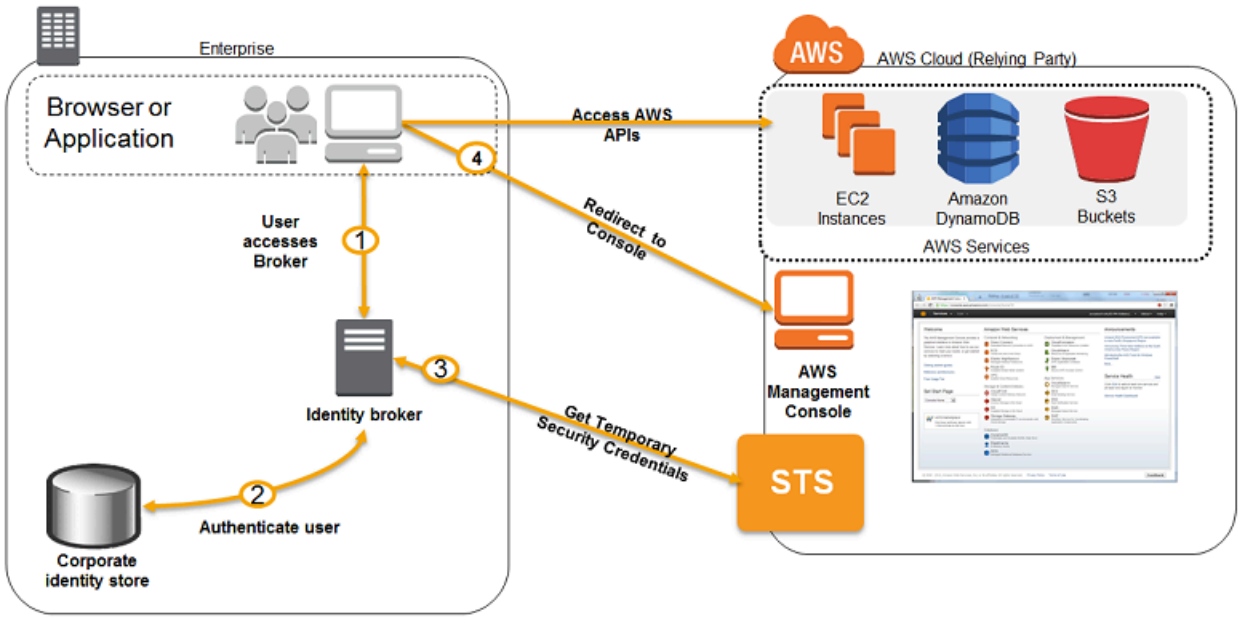
Wenn Ihre Identitätsquelle nicht mit SAML 2.0 kompatibel ist, können Sie eine benutzerdefinierte Identity Broker-Anwendung erstellen, um eine ähnliche Funktion durchzuführen. Die Brokeranwendung authentifiziert Benutzer, fordert temporäre Anmeldeinformationen für Benutzer an und stellt sie dann dem AWS Benutzer für den Zugriff auf Ressourcen zur Verfügung. AWS

Beispielsweise hat Example Corp. viele Mitarbeiter, die interne Anwendungen ausführen müssen, die auf die Ressourcen des AWS Unternehmens zugreifen. Die Mitarbeiter verfügen bereits über Identitäten im Identitäts- und Authentifizierungssystem des Unternehmens und das Beispielunternehmen möchte keinen separaten IAM-Benutzer für jeden Mitarbeiter erstellen.

Bob ist Entwickler bei Example Corp. Um internen Anwendungen von Example Corp. den Zugriff auf die AWS Ressourcen des Unternehmens zu ermöglichen, entwickelt Bob eine benutzerdefinierte

Identity Broker-Anwendung. Die Anwendung überprüft, ob Mitarbeiter beim vorhandenen Identitäts- und Authentifizierungssystem des Beispielunternehmens angemeldet sind, das möglicherweise LDAP, Active Directory oder ein anderes System nutzt. Die Identity Broker-Anwendung ruft dann die temporären Anmeldeinformationen für die Mitarbeiter ab. Dieses Szenario ähnelt dem vorherigen Szenario (eine mobile App, die ein benutzerdefiniertes Authentifizierungssystem verwendet), mit der Ausnahme, dass die Anwendungen, die Zugriff auf AWS Ressourcen benötigen, alle innerhalb des Unternehmensnetzwerks ausgeführt werden und das Unternehmen über ein vorhandenes Authentifizierungssystem verfügt.

Zum Abrufen von temporären Sicherheitsanmeldeinformationen ruft die Identity Broker-Anwendung entweder `AssumeRole` oder `GetFederationToken` auf, je nachdem, wie Bob die Richtlinien für die Benutzer verwalten möchte und wann die temporären Anmeldeinformationen ablaufen sollen. (Weitere Informationen zu den Unterschieden zwischen diesen API-Operationen finden Sie unter Temporäre IAM Sicherheitsanmeldeinformationen und Steuern von Berechtigungen für temporäre Sicherheitsanmeldeinformationen.) Der Aufruf gibt temporäre Sicherheitsanmeldeinformationen zurück, die aus einer AWS Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sitzungstoken bestehen. Die Identity Broker-Anwendung stellt diese temporären Sicherheitsanmeldeinformationen der internen Unternehmensanwendung zur Verfügung. Die Anwendung kann dann die temporären Anmeldeinformationen für direkte Aufrufe an AWS verwenden. Die Anwendung speichert die Anmeldeinformationen bis zum Ablaufdatum zwischen und fordert dann einen neuen Satz temporärer Anmeldeinformationen an. Die folgende Abbildung veranschaulicht dieses Szenario.



Dieses Szenario hat folgende Attribute:

- Die Identity Broker-Anwendung hat die Berechtigung, auf die STS-API (Security Token Service) von IAM zuzugreifen, um temporäre Sicherheitsanmeldeinformationen zu erstellen.
- Die Identity Broker-Anwendung kann überprüfen, ob die Mitarbeiter im vorhandenen Authentifizierungssystem authentifiziert sind.
- Benutzer können eine temporäre URL abrufen, über die sie auf die AWS Managementkonsole zugreifen können (was als Single Sign-On bezeichnet wird).

Weitere Informationen zum Erstellen von temporären Sicherheitsanmeldeinformationen finden Sie unter Anfordern temporärer Sicherheitsanmeldeinformationen. Weitere Informationen zu Verbundbenutzern, die Zugriff auf die AWS Management Console erhalten, finden Sie unter. Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console

Verwenden von serviceverknüpften Rollen

Eine serviceverknüpfte Rolle ist ein spezieller Typ von IAM-Rolle, der direkt mit einem AWS -Service verknüpft ist. Dienstbezogene Rollen sind vom Dienst vordefiniert und enthalten alle Berechtigungen, die der Dienst benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen. Der verknüpfte Service definiert auch, wie Sie eine serviceverknüpfte Rolle erstellen, ändern und löschen. Ein Service kann die Rolle automatisch erstellen oder löschen. Er kann Ihnen erlauben, die Rolle als Teil eines Assistenten oder Prozesses im Service zu erstellen, zu ändern oder zu löschen. Oder es kann erforderlich sein, dass Sie IAM zum Erstellen oder Löschen der Rolle verwenden. Unabhängig von der Methode erleichtern serviceverknüpfte Rollen den Prozess der Einrichtung eines Services, da Sie dem Service keine manuellen Berechtigungen für die Ausführung von Aktionen in Ihrem Namen hinzufügen müssen.

> **Note**
>
> Denken Sie daran, dass sich Servicerollen von serviceverknüpften Rollen unterscheiden. 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. 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. 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.

Der verknüpfte Service definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur dieser Service die Rollen zuordnen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Bevor Sie die Rollen löschen können, müssen Sie zunächst die zugehörigen Ressourcen löschen. Dies schützt Ihre -Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.

Tip

Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Berechtigungen von serviceverknüpften Rollen

Sie müssen Berechtigungen für eine IAM-Entität konfigurieren (z. B. Benutzer oder Rolle), damit der Benutzer eine serviceverknüpfte Rolle erstellen oder bearbeiten kann.

Note

Der ARN für eine serviceverknüpfte Rolle enthält einen Dienstauftraggeber, der in den nachfolgenden Richtlinien als *SERVICE-NAME*`.amazonaws.com` angegeben ist. Versuchen Sie nicht, den Dienstprinzipal zu erraten, da er zwischen Groß- und Kleinschreibung unterscheidet und das Format je nach AWS Dienst variieren kann. Um den Dienstauftraggeber für einen Service zu sehen, sehen Sie sich seine serviceverknüpfte Rollendokumentation an.

So erlauben Sie einer IAM-Entität das Erstellen einer bestimmten serviceverknüpften Rolle

Fügen Sie die folgende Richtlinie der IAM-Entität hinzu, um die serviceverknüpfte Rolle zu erstellen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

So erlauben Sie einer IAM-Entität das Erstellen einer beliebigen serviceverknüpften Rolle

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, um eine serviceverknüpfte Rolle oder eine beliebige Servicerolle zu erstellen, die die benötigten Richtlinien enthält. Diese Richtlinienanweisung gewährt der IAM-Entität nicht, der Rolle eine Richtlinie anzufügen.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

So erlauben Sie einer IAM-Entität das Bearbeiten der Beschreibung von beliebigen Servicerollen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, um die Beschreibung einer serviceverknüpften Rolle oder einer beliebigen Servicerolle zu bearbeiten.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

So erlauben Sie einer IAM-Entität das Löschen einer bestimmten serviceverknüpften Rolle

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die die serviceverknüpfte Rolle löschen soll.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

So erlauben Sie einer IAM-Entität das Löschen einer beliebigen serviceverknüpften Rolle

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die eine serviceverknüpfte Rolle löschen muss, aber keine Servicerolle.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

So erlauben Sie einer IAM-Entität, eine vorhandene Rolle den Service zu übergeben

Bei einigen AWS Diensten können Sie eine bestehende Rolle an den Dienst übergeben, anstatt eine neue dienstbezogene Rolle zu erstellen. Um dies zu tun, benötigt ein Benutzer Berechtigungen für das Übergeben der Rolle an den Service. Fügen Sie die folgende Anweisung

der Berechtigungsrichtlinie für die IAM-Entität hinzu, die eine Rolle übergeben soll. Diese Richtlinienanweisung ermöglicht außerdem der Entität, eine Liste der Rollen anzuzeigen, aus denen sie die zu übergebende Rolle auswählen kann. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Indirekte Berechtigungen mit serviceverknüpfte Rollen

Die Berechtigungen einer serviceverknüpften Rolle können indirekt auf andere Benutzer und Rollen übertragen werden. Wenn eine dienstverknüpfte Rolle von einem AWS Dienst verwendet wird, kann diese dienstverknüpfte Rolle ihre eigenen Berechtigungen verwenden, um andere Dienste aufzurufen. AWS Das bedeutet, dass Benutzer und Rollen mit Berechtigungen zum Aufrufen eines Services, der eine serviceverknüpfte Rolle verwendet, möglicherweise indirekten Zugriff auf Services haben, auf die über diese serviceverknüpfte Rolle zugegriffen werden kann.

Wenn Sie beispielsweise eine Amazon-RDS-DB-Instance erstellen, wird automatisch eine serviceverknüpfte Rolle für RDS erstellt, falls noch keine vorhanden ist. Diese servicebezogene Rolle ermöglicht es RDS, Amazon EC2, Amazon SNS, Amazon CloudWatch Logs und Amazon Kinesis in Ihrem Namen aufzurufen. Wenn Sie Benutzern und Rollen in Ihrem Konto erlauben, RDS-Datenbanken zu ändern oder zu erstellen, können sie möglicherweise indirekt mit Amazon EC2-, Amazon SNS-, Amazon CloudWatch Logs-Protokollen und Amazon Kinesis Kinesis-Ressourcen interagieren, indem sie RDS aufrufen, da RDS seine dienstbezogene Rolle für den Zugriff auf diese Ressourcen verwenden würde.

Erstellen einer serviceverknüpften Rolle

Die Vorgehensweise zum Erstellen einer serviceverknüpften Rolle hängt vom Service ab. In einigen Fällen müssen Sie die serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie beispielsweise eine

bestimmte Aktion (z. B. das Erstellen einer Ressource) im Service durchführen, erstellt der Service die serviceverknüpfte Rolle eventuell für Sie. Wenn Sie über einen Service verwenden, bevor dieser serviceverknüpfte Rollen unterstützt, hat der Service die Rolle eventuell bereits automatisch in Ihrem Konto erstellt. Weitere Informationen hierzu finden Sie unter In meinem AWS -Konto wird eine neue Rolle angezeigt.

In anderen Fällen unterstützt der Service möglicherweise das manuelle Erstellen einer serviceverknüpften Rolle mit der Servicekonsole, einer API oder der CLI. Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Um zu erfahren, ob der Service das Erstellen der serviceverknüpften Rolle unterstützt, wählen Sie den Link Yes aus, um die Dokumentation zur serviceverknüpften Rolle dieses Services anzuzeigen.

Wenn der Service das Erstellen der Rolle nicht unterstützt, können Sie IAM verwenden, um die serviceverknüpfte Rolle zu erstellen.

> **Important**
>
> Serviceverknüpfte Rollen werden Ihrem Limit für IAM-Rollen in einem AWS-Konto zugerechnet. Wenn Sie Ihr Limit erreicht haben, können Sie jedoch trotzdem serviceverknüpfte Rollen in Ihrem Konto erstellen. Nur serviceverknüpfte Rollen können das Limit überschreiten.

Erstellen einer serviceverknüpften Rolle (Konsole)

Bevor Sie eine serviceverknüpfte Rolle in IAM erstellen, müssen Sie ermitteln, ob der verknüpfte Service serviceverknüpfte Rollen automatisch erstellt und ob Sie die Rolle über die Konsole, eine API oder die CLI des Services erstellen können.

So erstellen Sie eine serviceverknüpfte Rolle (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. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus. Wählen Sie anschließend Create role (Rolle erstellen) aus.
3. Wählen Sie den Rollentyp AWS -Service.

4. Wählen Sie den Anwendungsfall für den Service. Anwendungsfälle werden durch den Service definiert, damit die für den Service erforderliche Vertrauensrichtlinie enthalten ist. Wählen Sie anschließend Weiter.
5. Wählen Sie Berechtigungsrichtlinien aus, die Sie an die Rolle anfügen möchten. Je nach ausgewähltem Anwendungsfall führt der Service einen der folgenden Schritte aus:

   - Definieren der Berechtigungen für die Rolle.
   - Erlaubt Ihnen die Auswahl aus einem begrenzten Satz von Berechtigungen.
   - Erlaubt Ihnen die Auswahl aus beliebigen Berechtigungen.
   - Sie dürfen zu diesem Zeitpunkt noch keine Richtlinien auswählen. Erstellen Sie die Richtlinien später und fügen Sie diese dann an die Rolle an.

   Aktivieren Sie das Kontrollkästchen neben der Richtlinie, die der Rolle die gewünschten Berechtigungen zuweist, und wählen Sie dann Next (Weiter) aus.

   > **Note**
   >
   > Die festgelegten Berechtigungen sind für alle Entitäten verfügbar, die die Rolle verwenden. Standardmäßig hat eine Rolle keine Berechtigungen.

6. Für Role name (Rollenname) werden die Anpassungsmöglichkeiten für den Rollennamen durch den Service festgelegt. Wenn der Service den Namen der Rolle definiert, kann diese Option nicht bearbeitet werden. In anderen Fällen kann der Service ein Präfix für die Rolle festlegen und Ihnen das Eingeben eines optionalen Suffixes erlauben.

   Geben Sie, wenn möglich, ein Suffix für den Rollennamen ein, das dem Standardnamen hinzugefügt wird. Dieses Suffix hilft Ihnen, den Zweck dieser Rolle zu identifizieren. Rollennamen müssen innerhalb Ihres AWS -Kontos eindeutig sein. Es wird hierbei nicht zwischen Groß- und Kleinschreibung unterschieden. z. B. können Sie keine Rollen erstellen, die **`<service-linked-role-name>_SAMPLE`** bzw. **`<service-linked-role-name>_sample`** heißen. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach der Erstellung nicht bearbeitet werden.
7. (Optional:) Bearbeiten Sie für Description (Beschreibung) die Beschreibung der neuen serviceverknüpften Rolle.

8. Sie können während der Erstellung keine Tags an servicegebundene Rollen anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Erstellen einer serviceverknüpften Rolle (AWS CLI)

Bevor Sie eine serviceverknüpfte Rolle in IAM erstellen, müssen Sie ermitteln, ob der verknüpfte Service serviceverknüpfte Rollen automatisch erstellt und ob Sie die Rolle über die CLI des Services erstellen können. Wenn die Service-CLI nicht unterstützt wird, verwenden Sie IAM-Befehle zum Erstellen einer serviceverknüpften Rolle mit der Vertrauensrichtlinie und den enthaltenen Richtlinien, die der Service benötigt, um die Rolle zuzuweisen.

So erstellen Sie eine serviceverknüpfte Rolle (AWS CLI)

Führen Sie den folgenden Befehl aus:

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Erstellen einer serviceverknüpften Rolle (AWS -API)

Bevor Sie eine serviceverknüpfte Rolle in IAM erstellen, müssen Sie ermitteln, ob der verknüpfte Service serviceverknüpfte Rollen automatisch erstellt und ob Sie die Rolle über die API des Services erstellen können. Wenn die Service-API nicht unterstützt wird, können Sie die AWS API verwenden, um eine serviceverknüpfte Rolle mit der Vertrauensrichtlinie und den Inline-Richtlinien zu erstellen, die der Dienst benötigt, um die Rolle zu übernehmen.

Um eine dienstbezogene Rolle (AWS API) zu erstellen

Verwenden Sie den CreateServiceLinkedRole-API-Aufruf. Geben Sie in der Anforderung einen Servicenamen im Format *SERVICE_NAME_URL*`.amazonaws.com` an.

Zum Erstellen der serviceverknüpften Rolle Lex Bots verwenden Sie z. B. `lex.amazonaws.com`.

Bearbeiten einer serviceverknüpften Rolle

Die Vorgehensweise zum Bearbeiten einer serviceverknüpften Rolle hängt vom Service ab. Einige Services erlauben möglicherweise das Bearbeiten der Berechtigungen für eine serviceverknüpfte Rolle über die Servicekonsole, eine API oder die CLI. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach der Erstellung einer serviceverknüpften Rolle nicht

bearbeitet werden. Sie können die Beschreibung beliebiger Rollen über die IAM-Konsole, eine API oder die CLI bearbeiten.

Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Um zu erfahren, ob der Service das Bearbeiten der serviceverknüpften Rolle unterstützt, wählen Sie den Link Yes aus, um die Dokumentation zur serviceverknüpften Rolle dieses Services anzuzeigen.

Bearbeiten der Beschreibung einer serviceverknüpften Rolle (Konsole)

Sie können die IAM-Konsole für das Bearbeiten der Beschreibung einer serviceverknüpften Rolle verwenden.

So bearbeiten Sie die Beschreibung einer serviceverknüpften Rolle (Konsole)

1. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus.
2. Wählen Sie den Namen der zu ändernden Rolle.
3. Wählen Sie neben Role description ganz rechts Edit.
4. Geben Sie eine neue Beschreibung im Dialogfeld ein und klicken Sie auf Save (Speichern).

Bearbeiten der Beschreibung einer serviceverknüpften Rolle (AWS CLI)

Sie können IAM-Befehle aus dem verwenden, AWS CLI um die Beschreibung einer serviceverknüpften Rolle zu bearbeiten.

So ändern Sie die Beschreibung einer serviceverknüpften Rolle (AWS CLI)

1. (Optional) Um die aktuelle Beschreibung einer Rolle anzuzeigen, führen Sie die folgenden Befehle aus:

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Verwenden Sie den Rollennamen, nicht den ARN, um sich auf Rollen mit den CLI-Befehlen zu beziehen. Wenn eine Rolle zum Beispiel folgenden ARN hat: `arn:aws:iam::123456789012:role/myrole`, verweisen Sie auf die Rolle als **`myrole`**.
2. Um die Beschreibung einer serviceverknüpften Rolle zu aktualisieren, führen Sie den folgenden Befehl aus:

```
aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
```

Bearbeiten einer Beschreibung einer serviceverknüpften Rolle (API)AWS

Sie können die AWS API verwenden, um die Beschreibung einer dienstbezogenen Rolle zu bearbeiten.

Um die Beschreibung einer dienstbezogenen Rolle (AWS API) zu ändern

1. (Optional:) Um die aktuelle Beschreibung einer Rolle anzuzeigen, rufen Sie die folgende Operation auf und geben den Namen der Rolle an:

   AWS API: GetRole
2. Um die Beschreibung einer Rolle zu ändern, rufen Sie die folgende Operation auf und geben den Namen (sowie optional eine Beschreibung) der Rolle an:

   AWS API: UpdateRole

Löschen einer serviceverknüpften Rolle

Die Vorgehensweise zum Erstellen einer serviceverknüpften Rolle hängt vom Service ab. In einigen Fällen müssen Sie die serviceverknüpfte Rolle nicht manuell löschen. Wenn Sie beispielsweise eine bestimmte Aktion (z. B. das Entfernen einer Ressource) im Service durchführen, löscht der Service die serviceverknüpfte Rolle eventuell für Sie.

In anderen Fällen unterstützt der Service möglicherweise das manuelle Löschen einer serviceverknüpften Rolle mit der Servicekonsole, einer API oder der AWS CLI.

Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Um zu erfahren, ob der Service das Löschen der serviceverknüpften Rolle unterstützt, wählen Sie den Link Yes aus, um die Dokumentation zur serviceverknüpften Rolle dieses Services anzuzeigen.

Wenn der Dienst das Löschen der Rolle nicht unterstützt, können Sie die mit dem Dienst verknüpfte Rolle aus der IAM-Konsole, API oder löschen. AWS CLI Wenn Sie ein Feature oder einen Service, die bzw. der eine serviceverknüpfte Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle

löschen. Auf diese Weise haben Sie keine ungenutzte Entität, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor Sie mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie sich zunächst vergewissern, dass die Rolle über keine aktiven Sitzungen verfügt, und alle Ressourcen entfernen, die von der Rolle verwendet werden.

So überprüfen Sie in der IAM-Konsole, ob die serviceverknüpfte Rolle über eine aktive Sitzung verfügt

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 der IAM Console Roles (Rollen) aus. Wählen Sie dann den Namen (nicht das Kontrollkästchen) der serviceverknüpften Rolle.
3. Wählen Sie auf der Seite Summary für die ausgewählte Rolle die Registerkarte Access Advisor.
4. Überprüfen Sie auf der Registerkarte Access Advisor die jüngsten Aktivitäten für die serviceverknüpfte Rolle.

> **Note**
>
> Wenn Sie sich nicht sicher sind, ob der Service die serviceverknüpfte 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 Ressourcen, die von einer serviceverknüpften Rolle verwendet werden

Informationen darüber, welche Services die Verwendung von serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Um zu erfahren, ob der Service das Löschen der serviceverknüpften Rolle unterstützt, wählen Sie den Link Yes aus, um die Dokumentation zur serviceverknüpften Rolle dieses Services anzuzeigen. Weitere Informationen zum Entfernen

von Ressourcen, die von einer serviceverknüpften Rolle verwendet werden, finden Sie in der Dokumentation des jeweiligen Services.

Löschen einer serviceverknüpften Rolle (Konsole)

Sie können die IAM-Konsole für das Löschen einer serviceverknüpften Rolle verwenden.

So löschen Sie eine serviceverknüpfte Rolle (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. Wählen Sie im Navigationsbereich der IAM Console Roles aus. Aktivieren Sie dann das Kontrollkästchen neben dem Rollennamen, den Sie löschen möchten, nicht den Namen oder die Zeile selbst.
3. Wählen Sie für Role actions oben auf der Seite Delete role.
4. Überprüfen Sie im Bestätigungsdialogfeld die Informationen, auf die zuletzt zugegriffen wurde. Darin wird angegeben, wann jede der ausgewählten Rollen zuletzt auf einen AWS Dienst zugegriffen hat. Auf diese Weise können Sie leichter bestätigen, ob die Rolle derzeit aktiv ist. Wenn Sie fortfahren möchten, wählen Sie Yes, Delete aus, um die serviceverknüpfte Rolle zur Löschung zu übermitteln.
5. Sehen Sie sich die Benachrichtigungen in der IAM-Konsole an, um den Fortschritt der Löschung der serviceverknüpften Rolle zu überwachen. Da die Löschung der serviceverknüpften IAM-Rolle asynchron erfolgt, kann die Löschung nach dem Übermitteln der Rolle für die Löschung erfolgreich sein oder fehlschlagen.

   - Wenn der Vorgang erfolgreich ist, wird die Rolle aus der Liste entfernt und eine Benachrichtigung des Erfolgs oben auf der Seite angezeigt.
   - Wenn der Vorgang fehlschlägt, können Sie in den Benachrichtigungen View details oder View Resources auswählen, um zu erfahren, warum die Löschung fehlgeschlagen ist. Wenn die Löschung fehlschlägt, weil die Rolle Ressourcen des Services verwendet, enthält die Benachrichtigung eine Liste der Ressourcen – sofern der Service diese Informationen zurückgibt. Sie können dann die Ressourcen bereinigen und die Löschung erneut durchführen.

     > **Note**
     >
     > Möglicherweise müssen Sie diesen Vorgang abhängig von den Informationen, die der Service zurückgibt, mehrmals wiederholen. Ihre serviceverknüpfte Rolle könnte beispielsweise sechs Ressourcen verwenden und Ihr Service Informationen zu fünf

davon zurückgeben. Wenn Sie die fünf Ressourcen bereinigen und die Rolle erneut für den Löschvorgang übermitteln, schlägt die Löschung fehl und der Service gibt die eine verbleibende Ressource zurück. Ein Service kann alle Ressourcen zurückgeben oder nur einige bzw. gar keine.

- Wenn die Aufgabe fehlschlägt und die Benachrichtigung keine Liste der Ressourcen enthält, gibt der Service möglicherweise diese Informationen nicht zurück. Informationen zum Bereinigen von Ressourcen für einen solchen Service finden Sie unter AWS Dienste, die mit IAM funktionieren. Um die Dokumentation der serviceverknüpften Rolle für den Service anzuzeigen, suchen Sie Ihren Service in der Tabelle und wählen Sie den Link Yes.

Löschen einer serviceverknüpften Rolle (AWS CLI)

Sie können IAM-Befehle von verwenden, AWS CLI um eine dienstverknüpfte Rolle zu löschen.

So löschen Sie eine serviceverknüpfte Rolle (AWS CLI)

1. Wenn Sie den Namen der serviceverknüpften Rolle, die Sie löschen möchten, nicht kennen, geben Sie den folgenden Befehl ein, um die Rollen in Ihrem Konto aufzulisten:

   ```
   aws iam get-role --role-name role-name
   ```

   Verwenden Sie den Rollennamen, nicht den ARN, um sich auf Rollen mit den CLI-Befehlen zu beziehen. Wenn eine Rolle zum Beispiel folgenden ARN hat: `arn:aws:iam::123456789012:role/myrole`, verweisen Sie auf die Rolle als **`myrole`**.

2. Da eine serviceverknüpfte Rolle nicht gelöscht werden kann, wenn sie verwendet wird oder ihr Ressourcen zugeordnet sind, müssen Sie eine Löschungsanforderung übermitteln. Diese Anforderung kann verweigert werden, wenn diese Bedingungen nicht erfüllt sind. Sie benötigen die `deletion-task-id` aus der Antwort, um den Status der Löschaufgabe zu überprüfen. Geben Sie den folgenden Befehl ein, um eine Löschanforderung für eine serviceverknüpfte Rolle zu übermitteln:

   ```
   aws iam delete-service-linked-role --role-name role-name
   ```

3. Geben Sie den folgenden Befehl ein, um den Status der Löschaufgabe zu überprüfen:

   ```
   aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

Der Status der Löschaufgabe kann `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` oder `FAILED` lauten. Wenn die Löschung fehlschlägt, gibt der Aufruf den Grund zurück, sodass Sie das Problem beheben können. Wenn die Löschung fehlschlägt, weil die Rolle Ressourcen des Services verwendet, enthält die Benachrichtigung eine Liste der Ressourcen – sofern der Service diese Informationen zurückgibt. Sie können dann die Ressourcen bereinigen und die Löschung erneut durchführen.

> **Note**
>
> Möglicherweise müssen Sie diesen Vorgang abhängig von den Informationen, die der Service zurückgibt, mehrmals wiederholen. Ihre serviceverknüpfte Rolle könnte beispielsweise sechs Ressourcen verwenden und Ihr Service Informationen zu fünf davon zurückgeben. Wenn Sie die fünf Ressourcen bereinigen und die Rolle erneut für den Löschvorgang übermitteln, schlägt die Löschung fehl und der Service gibt die eine verbleibende Ressource zurück. Ein Service kann alle Ressourcen zurückgeben oder nur einige bzw. gar keine. Informationen zum Bereinigen von Ressourcen für einen Service, der keine Ressourcen meldet, finden Sie unter AWS Dienste, die mit IAM funktionieren. Um die Dokumentation der serviceverknüpften Rolle für den Service anzuzeigen, suchen Sie Ihren Service in der Tabelle und wählen Sie den Link Yes.

Löschen einer serviceverknüpften Rolle (AWS -API)

Sie können die AWS API verwenden, um eine dienstverknüpfte Rolle zu löschen.

Um eine dienstverknüpfte Rolle (AWS API) zu löschen

1. Rufen Sie an, um eine Löschanfrage für eine dienstverknüpfte Rolle einzureichen. DeleteServiceLinkedRole Geben Sie in der Anforderung einen Rollennamen an.

   Da eine serviceverknüpfte Rolle nicht gelöscht werden kann, wenn sie verwendet wird oder ihr Ressourcen zugeordnet sind, müssen Sie eine Löschungsanforderung übermitteln. Diese Anforderung kann verweigert werden, wenn diese Bedingungen nicht erfüllt sind. Sie benötigen die `DeletionTaskId` aus der Antwort, um den Status der Löschaufgabe zu überprüfen.
2. Rufen GetServiceLinkedRoleDeletionStatusSie an, um den Status der Löschung zu überprüfen. Geben Sie in der Anforderung die `DeletionTaskId` an.

Der Status der Löschaufgabe kann `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` oder `FAILED` lauten. Wenn die Löschung fehlschlägt, gibt der Aufruf den Grund zurück, sodass Sie das Problem beheben können. Wenn die Löschung fehlschlägt, weil die Rolle Ressourcen des Services verwendet, enthält die Benachrichtigung eine Liste der Ressourcen – sofern der Service diese Informationen zurückgibt. Sie können dann die Ressourcen bereinigen und die Löschung erneut durchführen.

> **Note**
>
> Möglicherweise müssen Sie diesen Vorgang abhängig von den Informationen, die der Service zurückgibt, mehrmals wiederholen. Ihre serviceverknüpfte Rolle könnte beispielsweise sechs Ressourcen verwenden und Ihr Service Informationen zu fünf davon zurückgeben. Wenn Sie die fünf Ressourcen bereinigen und die Rolle erneut für den Löschvorgang übermitteln, schlägt die Löschung fehl und der Service gibt die eine verbleibende Ressource zurück. Ein Service kann alle Ressourcen zurückgeben oder nur einige bzw. gar keine. Informationen zum Bereinigen von Ressourcen für einen Service, der keine Ressourcen meldet, finden Sie unter AWS Dienste, die mit IAM funktionieren. Um die Dokumentation der serviceverknüpften Rolle für den Service anzuzeigen, suchen Sie Ihren Service in der Tabelle und wählen Sie den Link Yes.

Erstellen von IAM-Rollen

Um eine Rolle zu erstellen, können Sie die AWS Management Console AWS CLI, die Tools für Windows oder die PowerShell IAM-API verwenden.

Wenn Sie die verwenden AWS Management Console, führt Sie ein Assistent durch die Schritte zum Erstellen einer Rolle. Der Assistent hat leicht unterschiedliche Schritte, je nachdem, ob Sie eine Rolle für einen AWS Dienst, für einen AWS-Konto oder für einen Verbundbenutzer erstellen.

Themen

- Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer
- Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS -Service
- Erstellen von Rollen für externe Identitätsanbieter (Verbund)
- Erstellen einer Rolle mithilfe benutzerdefinierter Vertrauensrichtlinien (Konsole)
- Beispiele für Richtlinien zum Delegieren des Zugriffs

Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer

Sie können IAM-Rollen verwenden, um den Zugriff auf Ihre AWS Ressourcen zu delegieren. Mit IAM-Rollen können Sie Vertrauensbeziehungen zwischen Ihrem vertrauenswürdigen Konto und anderen vertrauenswürdigen Konten einrichten. AWS Das vertrauende Konto besitzt die Ressource, auf die zugegriffen werden soll, und das vertrauenswürdige Konto enthält die Benutzer, die Zugriff auf die Ressource erhalten müssen. Es ist jedoch möglich, dass ein anderes Konto eine Ressource in Ihrem Konto besitzt. Beispielsweise kann das Vertrauenskonto dem vertrauenswürdigen Konto erlauben, neue Ressourcen zu erstellen, z. B. neue Objekte in einem Amazon-S3-Bucket. In diesem Fall besitzt das Konto, das die Ressource erstellt, die Ressource und steuert, wer auf sie zugreifen kann.

Nachdem Sie die Vertrauensstellung erstellt haben, kann ein IAM-Benutzer oder eine Anwendung aus dem vertrauenswürdigen Konto den API-Vorgang AWS Security Token Service (AWS STS) `AssumeRole`verwenden. Dieser Vorgang stellt temporäre Sicherheitsanmeldedaten bereit, die den Zugriff auf AWS Ressourcen in Ihrem Konto ermöglichen.

Sie können die beiden Konten steuern oder das Konto mit den Benutzern kann von einem Drittanbieter gesteuert werden. Wenn es sich bei dem anderen Konto mit den Benutzern um ein Konto handelt, AWS-Konto das Sie nicht kontrollieren, können Sie das `externalId` Attribut verwenden. Die externe ID kann ein beliebiges Wort oder eine beliebige Zahl sein, das bzw. die zwischen Ihnen und dem Administrator des Drittanbieter-Kontos vereinbart wurde. Durch diese Option wird der Vertrauensrichtlinie automatisch eine Bedingung hinzugefügt, mit der der Benutzer die Rolle nur dann übernehmen kann, wenn die Anforderung die richtige `sts:ExternalID` enthält. Weitere Informationen finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.

Weitere Informationen zur Verwendung von Rollen zum Delegieren von Berechtigungen finden Sie unter Rollenbegriffe und -konzepte. Informationen zur Verwendung einer Servicerolle zum Erlauben des Zugriffs auf Ressourcen in Ihrem Konto durch Services finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS -Service.

Erstellen einer IAM-Rolle (Konsole)

Sie können das verwenden AWS Management Console , um eine Rolle zu erstellen, die ein IAM-Benutzer übernehmen kann. Gehen Sie beispielsweise davon aus, dass Ihr Unternehmen über mehrere Systeme verfügt AWS-Konten , um eine Entwicklungsumgebung von einer Produktionsumgebung zu isolieren. Allgemeine Informationen zum Erstellen einer Rolle, mit der Benutzer im Entwicklungskonto auf Ressourcen in der Produktionsumgebung zugreifen können, finden Sie unter Beispielszenario mit getrennten Entwicklungs- und Produktionskonten.

So erstellen Sie eine Rolle (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 Konsole auf Roles (Rollen) und wählen Sie dann Create role (Rolle erstellen).
3. Wählen Sie den Rollentyp AWS-Konto.
4. Um eine Rolle für Ihr Konto zu erstellen, wählen Sie This account (Dieses Konto) aus. Um eine Rolle für ein anderes Konto zu erstellen, wählen Sie Anderes AWS-Konto und geben Sie die Konto-ID ein, der Sie Zugriff auf Ihre Ressourcen gewähren möchten.

   Der Administrator des angegebenen Kontos kann die Berechtigung erteilen, diese Rolle für alle IAM-Benutzer in diesem Konto zu übernehmen. Hierzu fügt der Administrator eine Richtlinie an den Benutzer oder eine Gruppe an, mit der die Berechtigung für die Aktion `sts:AssumeRole` gewährt wird. Diese Richtlinie muss den ARN der Rolle als `Resource` angeben.
5. Wenn Sie Benutzern Berechtigungen von einem Konto gewähren, das Sie nicht kontrollieren, und die Benutzer diese Rolle programmgesteuert übernehmen, wählen Sie Require external ID (Externe ID fordern). Die externe ID kann ein beliebiges Wort oder eine beliebige Zahl sein, das bzw. die zwischen Ihnen und dem Administrator des Drittanbieter-Kontos vereinbart wurde. Durch diese Option wird der Vertrauensrichtlinie automatisch eine Bedingung hinzugefügt, mit der der Benutzer die Rolle nur dann übernehmen kann, wenn die Anforderung die richtige `sts:ExternalID` enthält. Weitere Informationen finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.

   > **Important**
   >
   > Wenn Sie diese Option wählen, wird der Zugriff auf die Rolle nur über die AWS CLI Tools für Windows oder die PowerShell AWS API eingeschränkt. Das liegt daran, dass Sie die AWS Konsole nicht verwenden können, um zu einer Rolle zu wechseln, deren Vertrauensrichtlinie eine `externalId` Bedingung enthält. Sie können jedoch diese Art des Zugriffs programmgesteuert erstellen, indem Sie mithilfe des relevanten SDK ein Skript oder eine Anwendung schreiben. Weitere Informationen und ein Beispielskript finden Sie unter How to Enable Cross-Account Access to the AWS Management Console im AWS Sicherheits-Blog.
6. Wenn Sie die Rolle auf Benutzer beschränken möchten, die sich über Multifaktor-Authentifizierung (MFA) anmelden, wählen Sie Require MFA (MFA erforderlich). Dadurch wird

eine Bedingung zur Vertrauensrichtlinie der Rolle hinzugefügt, mit der geprüft wird, ob eine MFA-Anmeldung vorliegt. Ein Benutzer, der die Rolle übernehmen möchte, muss sich über ein konfiguriertes MFA-Gerät mit einem temporären einmaligen Passwort anmelden. Benutzer ohne MFA-Authentifizierung können die Rolle nicht übernehmen. Weitere Informationen zu MFA finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

7. Wählen Sie Weiter aus.
8. IAM enthält eine Liste der AWS verwalteten und kundenverwalteten Richtlinien in Ihrem Konto. Wählen Sie die Richtlinie aus, die für die Berechtigungsrichtlinie verwendet werden soll, oder wählen Create policy (Richtlinie erstellen), um eine neue Registerkarte im Browser zu öffnen und eine vollständig neue Richtlinie zu erstellen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien. Nachdem Sie die Richtlinie erstellt haben, schließen Sie die Registerkarte und kehren zur ursprünglichen Registerkarte zurück. Aktivieren Sie die Kontrollkästchen neben den Berechtigungsrichtlinien, die jeder, der die Rolle übernimmt, erhalten soll. Wenn Sie es vorziehen, zu diesem Zeitpunkt keine Richtlinien auszuwählen, können Sie sie der Rolle später hinzufügen. Standardmäßig hat eine Rolle keine Berechtigungen.
9. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature.

   Öffnen Sie den Abschnitt Set permissions boundary (Berechtigungsgrenze festlegen) und wählen Sie Use a permissions boundary to control the maximum role permissions (Eine Berechtigungsgrenze verwenden, um die maximalen Rollenberechtigungen zu steuern). Wählen Sie die Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll.
10. Wählen Sie Weiter aus.
11. Geben Sie unter Role name (Rollenname) einen Namen für Ihre Rolle ein. Rollennamen müssen innerhalb Ihres AWS-Konto Unternehmens eindeutig sein. Wenn ein Rollenname in einer Richtlinie oder als Teil eines ARN verwendet wird, muss die Groß-/Kleinschreibung des Rollennamens beachtet werden. Wenn Kunden in der Konsole ein Rollenname angezeigt wird, beispielsweise während des Anmeldevorgangs, wird die Groß-/Kleinschreibung des Rollennamens nicht beachtet. Da verschiedene Entitäten möglicherweise auf die Rolle verweisen, können Sie den Namen der Rolle nach der Erstellung nicht mehr bearbeiten.
12. (Optional) Geben Sie im Feld Description (Beschreibung) eine Beschreibung für die neue Rolle ein.
13. Wählen Sie in den Abschnitten Step 1: Select trusted entities (Schritt 1: Vertrauenswürdige Entitäten auswählen) oder Step 2: Add permissions (Schritt 2: Berechtigungen hinzufügen) die Option Edit (Bearbeiten) aus, um die Anwendungsfälle und Berechtigungen für die Rolle

zu bearbeiten. Sie werden zu den vorherigen Seiten zurückgeleitet, um die Änderungen vorzunehmen.

14. (Optional) Fügen Sie der Rolle Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
15. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

> **Important**
>
> Dies ist allerdings nur der erste Teil der erforderlichen Konfiguration. Sie müssen auch einzelnen Benutzern in den vertrauenswürdigen Konten die Berechtigung geben, in die Rolle in der Konsole zu wechseln oder die Rolle programmgesteuert zu übernehmen. Weitere Informationen zu diesem Schritt finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

Erstellen einer IAM-Rolle (AWS CLI)

Das Erstellen einer Rolle aus dem AWS CLI umfasst mehrere Schritte. Wenn Sie die Konsole verwenden, um eine Rolle zu erstellen, werden viele der Schritte für Sie erledigt, aber mit der müssen AWS CLI Sie jeden Schritt explizit selbst ausführen. Sie müssen die Rolle erstellen und ihr dann eine Berechtigungsrichtlinie zuweisen. Optional können Sie auch die Berechtigungsgrenze für Ihre Rolle festlegen.

So erstellen Sie eine Rolle für kontoübergreifenden Zugriff (AWS CLI)

1. Erstellen Sie eine Rolle: aws iam create-role
2. Fügen Sie der Rolle eine Richtlinie für verwaltete Berechtigungen hinzu: aws iam attach-role-policy

   or

   Erstellen Sie eine Inline-Berechtigungsrichtlinie für die Rolle: aws iam put-role-policy
3. (Optional) Fügen Sie der Rolle benutzerdefinierte Attribute durch Zuweisen von Tags hinzu: aws iam tag-role

   Weitere Informationen finden Sie unter Verwaltung von Tags in IAM-Rollen (AWS CLI oder AWS API).

4. (Optional) Legen Sie die Berechtigungsgrenze für die Rolle fest: aws iam put-role-permissions-boundary

   Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die eine Rolle haben kann. Bei den Berechtigungsgrenzen handelt es sich um eine erweiterte AWS Funktion.

Das folgende Beispiel zeigt die ersten beiden und häufigsten Schritte zum Anlegen einer kontenübergreifenden Rolle in einer einfachen Umgebung. Dieses Beispiel erlaubt jedem Benutzer im `123456789012`-Konto, die Rolle zu übernehmen und den `example_bucket`-Amazon S3 Bucket anzuzeigen. In diesem Beispiel wird auch davon ausgegangen, dass Sie einen Client-Computer mit Windows verwenden und bereits die Befehlszeilenschnittstelle mit Ihren Konto-Anmeldeinformationen und der Region konfiguriert haben. Weitere Informationen finden Sie unter Konfiguration der AWS Befehlszeilenschnittstelle.

In diesem Beispiel nehmen Sie die folgende Vertrauensrichtlinie in den ersten Befehl auf, wenn Sie die Rolle erstellen. Diese Vertrauensrichtlinie gestattet Benutzern im `123456789012`-Konto, die Rolle unter Verwendung der `AssumeRole`-Operation anzunehmen, aber nur, wenn der Benutzer über die Parameter `SerialNumber` und `TokenCode` eine MFA-Authentifizierung bereitstellt. Weitere Informationen zu MFA finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
          "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
      }
  ]
}
```

Important

Wenn Ihr `Principal`-Element einen ARN für eine bestimmte IAM-Rolle oder einen bestimmten IAM-Benutzer enthält, wird dieser ARN beim Speichern der Richtlinie in eine eindeutige Auftraggeber-ID umgewandelt wird. Auf diese Weise wird das Risiko reduziert, dass jemand seine Berechtigungen durch Entfernen und Neuerstellen der Rolle oder des Benutzers erweitert. Normalerweise wird diese ID nicht in der Konsole angezeigt, da bei der

Anzeige der Vertrauensrichtlinie auch eine Rückumwandlung zum ARN erfolgt. Wenn Sie die Rolle oder den Benutzer jedoch löschen, wird die Prinzipal-ID in der Konsole angezeigt, da sie nicht mehr einem ARN zugeordnet werden AWS kann. Wenn Sie also einen Benutzer oder eine Rolle löschen und neu erstellen, auf die beide im `Principal`-Element einer Vertrauensrichtlinie verwiesen wird, müssen Sie die Rolle bearbeiten, um den ARN zu ersetzen.

Wenn Sie den zweiten Befehl verwenden, müssen Sie der Rolle eine vorhandene verwaltete Richtlinie hinzufügen. Die folgende Berechtigungsrichtlinie erlaubt jedem, der die Rolle übernimmt, nur die Aktion `ListBucket` auf dem `example_bucket`-Amazon S3-Bucket durchzuführen.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
  ]
}
```

Um diese `Test-UserAccess-Role`-Rolle zu erstellen, müssen Sie zunächst die vorherige Vertrauensrichtlinie mit den Namen `trustpolicyforacct123456789012.json` im `policies`-Ordner in Ihrem lokalen `C:`-Laufwerk speichern. Speichern Sie dann die vorherige Berechtigungsrichtlinie als vom Kunden verwaltete Richtlinie in Ihrer AWS-Konto mit dem Namen`PolicyForRole`. Sie können dann die folgenden Befehle verwenden, um die Rolle zu erstellen und die verwaltete Richtlinie hinzuzufügen.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> Dies ist allerdings nur der erste Teil der erforderlichen Konfiguration. Sie müssen auch einzelnen Benutzern im vertrauenswürdigen Konto Berechtigungen zum Wechseln der Rolle erteilen. Weitere Informationen zu diesem Schritt finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

Nachdem Sie die Rolle erstellt und ihr Berechtigungen zur Ausführung von AWS Aufgaben oder zum Zugriff auf AWS Ressourcen erteilt haben, können alle Benutzer im `123456789012` Konto die Rolle übernehmen. Weitere Informationen finden Sie unter Wechseln zu einer IAM-Rolle (AWS CLI).

Eine IAM-Rolle (AWS API) erstellen

Das Erstellen einer Rolle über die AWS API umfasst mehrere Schritte. Wenn Sie eine Rolle mithilfe der Konsole erstellen, werden viele Schritte automatisch abgeschlossen. In API müssen Sie diese Schritte jedoch manuell ausführen. Sie müssen die Rolle erstellen und ihr dann eine Berechtigungsrichtlinie zuweisen. Optional können Sie auch die Berechtigungsgrenze für Ihre Rolle festlegen.

Um eine Rolle im Code (AWS API) zu erstellen

1. Eine Rolle erstellen: CreateRole

   Für die Vertrauensrichtlinie der Rolle können Sie einen Dateispeicherort angeben.
2. Fügen Sie der Rolle eine verwaltete Berechtigungsrichtlinie hinzu: AttachRolePolicy

   or

   Erstellen Sie eine Inline-Berechtigungsrichtlinie für die Rolle: PutRolePolicy

   > **Important**
   >
   > Dies ist allerdings nur der erste Teil der erforderlichen Konfiguration. Sie müssen auch einzelnen Benutzern im vertrauenswürdigen Konto Berechtigungen zum Wechseln der Rolle erteilen. Weitere Informationen zu diesem Schritt finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

3. (Optional) Fügen Sie dem Benutzer benutzerdefinierte Attribute hinzu, indem Sie Tags anhängen: TagRole

   Weitere Informationen finden Sie unter Verwaltung von Tags für IAM-Benutzer (AWS CLI oder AWS API).

4. (Optional) Legen Sie die Berechtigungsgrenze für die Rolle fest: PutRolePermissionsBoundary

   Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die eine Rolle haben kann. Bei den Berechtigungsgrenzen handelt es sich um eine erweiterte AWS Funktion.

Nachdem Sie die Rolle erstellt und ihr Berechtigungen zur Ausführung von AWS Aufgaben oder zum Zugriff auf AWS Ressourcen erteilt haben, müssen Sie den Benutzern im Konto Berechtigungen erteilen, damit sie die Rolle übernehmen können. Weitere Informationen zum Übernehmen einer Rolle finden Sie unter Zu einer IAM-Rolle (AWS API) wechseln.

Erstellen einer IAM-Rolle (AWS CloudFormation)

Informationen zum Erstellen einer IAM-Rolle in finden Sie in AWS CloudFormation der Ressourcen- und Eigenschaftenreferenz sowie in den Beispielen im AWS CloudFormation Benutzerhandbuch.

Weitere Informationen zu IAM-Vorlagen finden Sie unter AWS Identity and Access Management Vorlagenausschnitte im Benutzerhandbuch. AWS CloudFormationAWS CloudFormation

Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS -Service

Für viele AWS Dienste müssen Sie Rollen verwenden, damit der Dienst in Ihrem Namen auf Ressourcen in anderen Diensten zugreifen kann. Eine Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen durchzuführen, wird als Servicerolle bezeichnet. Wenn eine Rolle einem speziellen Zweck für einen Service dient, wird sie als Servicerolle für EC2-Instances (beispielsweise) oder als serviceverknüpfte Rolle kategorisiert. Informationen zu den Services, die serviceverknüpfte Rollen unterstützen, oder zu Services mit einer Form der temporären Anmeldeinformationen finden Sie unter AWS Dienste, die mit IAM funktionieren. Wenn Sie erfahren möchten, wie ein einzelner Service Rollen verwendet, wählen Sie den Servicenamen in der Tabelle aus, um die Dokumentation des jeweiligen Service anzuzeigen.

Wenn Sie die `PassRole` Berechtigung festlegen, sollten Sie sicherstellen, dass ein Benutzer keine Rolle weitergibt, bei der die Rolle über mehr Berechtigungen verfügt, als Sie dem Benutzer zuweisen möchten. Beispielsweise darf Alice möglicherweise keine Amazon S3 S3-Aktionen ausführen. Wenn

Alice eine Rolle an einen Service übergeben könnte, der Amazon S3 S3-Aktionen zulässt, könnte der Service bei der Ausführung des Jobs Amazon S3 S3-Aktionen im Namen von Alice ausführen.

Informationen darüber, wie Rollen Ihnen das Delegieren von Berechtigungen erleichtern, finden Sie unter Rollenbegriffe und -konzepte.

Servicerollen-Berechtigungen

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rolle) eine serviceverknüpfte Rolle erstellen oder bearbeiten kann.

> **Note**
>
> Der ARN für eine serviceverknüpfte Rolle enthält einen Service-Prinzipal, der in den folgenden Richtlinien als *SERVICE-NAME*`.amazonaws.com` angegeben ist. Versuchen Sie nicht, den Service-Prinzipal zu erraten, da es auf Groß- und Kleinschreibung ankommt und das Format je nach AWS -Services variieren kann. Um den Dienstauftraggeber für einen Service zu sehen, sehen Sie sich seine serviceverknüpfte Rollendokumentation an.

So erlauben Sie einer IAM-Entität das Erstellen einer bestimmten serviceverknüpften Rolle

Fügen Sie die folgende Richtlinie der IAM-Entität hinzu, um die Servicerolle zu erstellen. Mit dieser Richtlinie können Sie eine Servicerolle für den angegebenen Service und mit einem bestimmten Namen erstellen. Anschließend können Sie verwaltete oder Inline-Richtlinien an die Rolle anfügen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

So erlauben Sie einer IAM-Entität das Erstellen einer beliebigen Servicerolle

AWS empfiehlt, dass Sie nur administrativen Benutzern das Erstellen von Servicerollen gestatten. Eine Person mit Berechtigungen zum Erstellen einer Rolle und zum Anfügen einer Richtlinie kann ihre eigenen Berechtigungen eskalieren. Erstellen Sie stattdessen eine Richtlinie, die es ihnen ermöglicht, nur die Rollen zu erstellen, die sie benötigen. Oder lassen Sie einen Administrator die Servicerolle in ihrem Namen erstellen.

Verwenden Sie die AdministratorAccess AWS verwaltete Richtlinie, um eine Richtlinie anzuhängen, die es einem Administrator ermöglicht AWS-Konto, auf Ihre gesamte Richtlinie zuzugreifen.

So erlauben Sie einer IAM-Entität das Bearbeiten einer bestimmten Servicerolle

Fügen Sie die folgende Richtlinie der IAM-Entität hinzu, um die Servicerolle zu bearbeiten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

So erlauben Sie einer IAM-Entität das Löschen einer bestimmten Servicerolle

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, um die angegebene Servicerolle zu löschen.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

So erlauben Sie einer IAM-Entität das Löschen einer beliebigen Servicerolle

AWS empfiehlt, dass Sie nur Administratorbenutzern das Löschen von Servicerollen gestatten. Erstellen Sie stattdessen eine Richtlinie, mit der sie nur die Rollen löschen können, die sie benötigen, oder dass ein Administrator die Servicerolle in ihrem Namen löscht.

Verwenden Sie die AdministratorAccess AWS verwaltete Richtlinie, um eine Richtlinie anzuhängen, die es einem Administrator ermöglicht AWS-Konto, auf Ihre gesamte Richtlinie zuzugreifen.

Eine Rolle für einen AWS Dienst (Konsole) erstellen

Sie können den verwenden AWS Management Console , um eine Rolle für einen Dienst zu erstellen. Da einige Services mehrere Servicerollen unterstützen, finden Sie in der AWS -Dokumentation für Ihren Service Informationen dazu, welcher Anwendungsfall auszuwählen ist. Sie können erfahren, wie Sie der Rolle die erforderlichen Vertrauens- und Berechtigungsrichtlinien zuweisen, damit der Service die Rolle in Ihrem Namen zuordnen kann. Die Schritte, mit denen Sie die Berechtigungen für Ihre Rolle steuern können, hängen davon ab, wie der Service die Anwendungsfälle definiert und ob Sie eine servicegebundene Rolle anlegen.

Um eine Rolle für eine AWS-Service (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 Sie für Service oder Anwendungsfall einen Service und dann den Anwendungsfall aus. Anwendungsfälle werden durch den Service definiert, damit die für den Service erforderliche Vertrauensrichtlinie enthalten ist.
5. Wählen Sie Weiter aus.
6. Bei Berechtigungsrichtlinien hängen die Optionen vom ausgewählten Anwendungsfall ab:
   - Wenn der Dienst die Berechtigungen für die Rolle definiert, können Sie keine Berechtigungsrichtlinien auswählen.
   - Wählen Sie aus einer begrenzten Anzahl von Berechtigungsrichtlinien aus.
   - Wählen Sie aus allen Berechtigungsrichtlinien aus.
   - Wählen Sie keine Berechtigungsrichtlinien aus, erstellen Sie die Richtlinien, nachdem die Rolle erstellt wurde, und fügen Sie die Richtlinien dann der Rolle hinzu.
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. Die Optionen für den Rollennamen hängen vom Dienst ab:
   - Wenn der Dienst den Rollennamen definiert, können Sie den Rollennamen nicht bearbeiten.
   - Wenn der Dienst ein Präfix für den Rollennamen definiert, können Sie ein optionales Suffix eingeben.
   - Wenn der Dienst den Rollennamen nicht definiert, können Sie der Rolle einen Namen geben.

   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) Fügen Sie Tags als Schlüssel-Wert-Paare hinzu, um die Rolle leichter zu identifizieren, zu organisieren oder nach ihr zu suchen. 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).

Erstellen einer Rolle für einen Service (AWS CLI)

Das Erstellen einer Rolle aus der AWS CLI umfasst mehrere Schritte. Wenn Sie die Konsole verwenden, um eine Rolle zu erstellen, werden viele der Schritte für Sie erledigt, aber mit der müssen AWS CLI Sie jeden Schritt explizit selbst ausführen. Sie müssen die Rolle erstellen und ihr dann eine Berechtigungsrichtlinie zuweisen. Wenn der Service, mit dem Sie arbeiten, Amazon EC2 ist, müssen Sie außerdem ein Instance-Profil erstellen und die Rolle diesem Profil hinzufügen. Optional können Sie auch die Berechtigungsgrenze für Ihre Rolle festlegen.

Um eine Rolle für einen AWS Dienst aus dem zu erstellen AWS CLI

1. Mit dem folgenden Befehl `create-role` wird eine Rolle mit dem Namen Test-Rolle erstellt und ihr eine Vertrauensrichtlinie zugewiesen:

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. Fügen Sie der Rolle eine Richtlinie für verwaltete Berechtigungen hinzu: aws iam attach-role-policy.

Mit dem folgenden Befehl `attach-role-policy` wird beispielsweise die verwaltete AWS -Richtlinie namens `ReadOnlyAccess` an die IAM-Rolle namens `ReadOnlyRole` angehängt:

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
ReadOnlyAccess --role-name ReadOnlyRole
```

or

Erstellen Sie eine Inline-Berechtigungsrichtlinie für die Rolle: aws iam put-role-policy

Informationen zum Hinzufügen einer Inline-Berechtigungsrichtlinie finden Sie im folgenden Beispiel:

```
aws iam put-role-policy --role-name Test-Role --policy-name
ExamplePolicy --policy-document file://AdminPolicy.json
```

3. (Optional) Fügen Sie der Rolle benutzerdefinierte Attribute durch Zuweisen von Tags hinzu: aws iam tag-role

   Weitere Informationen finden Sie unter Verwaltung von Tags in IAM-Rollen (AWS CLI oder AWS API).

4. (Optional) Legen Sie die Berechtigungsgrenze für die Rolle fest: aws iam put-role-permissions-boundary

   Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die eine Rolle haben kann. Bei den Berechtigungsgrenzen handelt es sich um eine erweiterte AWS Funktion.

Wenn Sie die Rolle mit Amazon EC2 oder einem anderen AWS Service verwenden möchten, der Amazon EC2 verwendet, müssen Sie die Rolle in einem Instanzprofil speichern. Ein Instance-Profil ist ein Container für eine Rolle, der beim Start an eine Amazon EC2-Instance angefügt werden kann. Ein Instance-Profil kann nur eine -Rolle enthalten und dieses Limit kann nicht erhöht werden. Wenn Sie die Rolle mithilfe von erstellen AWS Management Console, wird das Instance-Profil mit demselben Namen wie die Rolle für Sie erstellt. Weitere Informationen zu Instance-Profilen finden Sie unter Verwenden von Instance-Profilen. Informationen zum Starten einer EC2-Instance mit einer Rolle finden Sie unter Steuern des Zugriffs auf Amazon EC2 EC2-Ressourcen im Amazon EC2 EC2-Benutzerhandbuch.

So erstellen Sie ein Instance-Profil und speichern die Rolle darin (AWS CLI)

1. Erstellen Sie ein Instance-Profil: aws iam create-instance-profile
2. Fügen Sie die Rolle dem Instanzprofil hinzu: aws add-role-to-instance iam -profile

Der folgende AWS CLI Beispielbefehlssatz veranschaulicht die ersten beiden Schritte zum Erstellen einer Rolle und zum Anhängen von Berechtigungen. Es zeigt auch die beiden Schritte zum Anlegen eines Instance-Profils und zum Hinzufügen der Rolle zum Profil. Diese Beispielvertrauensrichtlinie erlaubt dem Amazon EC2-Service, die Rolle zu übernehmen und den `example_bucket`– Amazon S3-Bucket anzuzeigen. In diesem Beispiel wird auch davon ausgegangen, dass Sie einen Client-Computer mit Windows ausführen und bereits die Befehlszeilenschnittstelle mit den Anmeldeinformationen und der Region für Ihr Konto konfiguriert haben. Weitere Informationen finden Sie unter Konfiguration der AWS Befehlszeilenschnittstelle.

In diesem Beispiel nehmen Sie die folgende Vertrauensrichtlinie in den ersten Befehl auf, wenn Sie die Rolle erstellen. Diese Vertrauensrichtlinie gestattet dem Amazon EC2-Service, die Rolle zu übernehmen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Wenn Sie den zweiten Befehl verwenden, müssen Sie der Rolle eine Berechtigungsrichtlinie hinzufügen. Die folgende Beispiel-Berechtigungsrichtlinie gestattet der Rolle nur, die `ListBucket`-Aktion für den `example_bucket` Amazon S3-Bucket auszuführen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Um diese `Test-Role-for-EC2`-Rolle zu erstellen, müssen Sie zuvor die vorhergehende Vertrauensrichtlinie mit dem Namen `trustpolicyforec2.json` und die vorherige Berechtigungsrichtlinie mit dem Namen `permissionspolicyforec2.json` im `policies`-Verzeichnis Ihres lokalen `C:`-Laufwerks speichern. Mit den folgenden Befehlen können Sie dann die Rolle erstellen, die Richtlinie hinzufügen, das Instance-Profil erstellen und die Rolle dem Instance-Profil hinzufügen.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Wenn Sie die EC2-Instance starten, geben Sie den Namen des Instance-Profils auf der Seite „Instanzdetails konfigurieren“ an, wenn Sie die AWS Konsole verwenden. Wenn Sie den CLI-Befehl `aws ec2 run-instances` verwenden, legen Sie den `--iam-instance-profile`-Parameter fest.

Erstellen einer Rolle für einen Service (AWS -API)

Das Erstellen einer Rolle über die AWS API umfasst mehrere Schritte. Wenn Sie eine Rolle mithilfe der Konsole erstellen, werden viele Schritte automatisch abgeschlossen. In API müssen Sie diese Schritte jedoch manuell ausführen. Sie müssen die Rolle erstellen und ihr dann eine Berechtigungsrichtlinie zuweisen. Wenn der Service, mit dem Sie arbeiten, Amazon EC2 ist, müssen Sie außerdem ein Instance-Profil erstellen und die Rolle diesem Profil hinzufügen. Optional können Sie auch die Berechtigungsgrenze für Ihre Rolle festlegen.

Um eine Rolle für einen AWS Dienst (AWS API) zu erstellen

1. Eine Rolle erstellen: CreateRole

Für die Vertrauensrichtlinie der Rolle können Sie einen Dateispeicherort angeben.

2. Fügen Sie der Rolle eine Richtlinie für verwaltete Berechtigungen hinzu: AttachRoleRichtlinie

   or

   Erstellen Sie eine Inline-Berechtigungsrichtlinie für die Rolle: PutRoleRichtlinie

3. (Optional) Fügen Sie dem Benutzer benutzerdefinierte Attribute hinzu, indem Sie Tags anhängen: TagRole

   Weitere Informationen finden Sie unter Verwaltung von Tags für IAM-Benutzer (AWS CLI oder AWS API).

4. (Optional) Legen Sie die Berechtigungsgrenze für die Rolle fest: PutRolePermissionsBoundary

   Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die eine Rolle haben kann. Bei den Berechtigungsgrenzen handelt es sich um eine erweiterte AWS Funktion.

Wenn Sie die Rolle mit Amazon EC2 oder einem anderen AWS Service verwenden möchten, der Amazon EC2 verwendet, müssen Sie die Rolle in einem Instanzprofil speichern. Ein Instance-Profil ist ein Container für eine Rolle. Jedes Instance-Profil kann nur eine Rolle enthalten und dieses Limit kann nicht erhöht werden. Wenn Sie die Rolle in der erstellen AWS Management Console, wird das Instance-Profil für Sie mit demselben Namen wie die Rolle erstellt. Weitere Informationen zu Instance-Profilen finden Sie unter Verwenden von Instance-Profilen. Informationen zum Starten einer Amazon EC2 EC2-Instance mit einer Rolle finden Sie unter Steuern des Zugriffs auf Amazon EC2 EC2-Ressourcen im Amazon EC2 EC2-Benutzerhandbuch.

Um ein Instance-Profil zu erstellen und die Rolle darin zu speichern (API)AWS

1. Erstellen Sie ein Instanzprofil: CreateInstanceProfil
2. Fügen Sie die Rolle dem Instanzprofil hinzu: AddRoleToInstanceProfil

Erstellen von Rollen für externe Identitätsanbieter (Verbund)

Sie können Identitätsanbieter verwenden, anstatt IAM-Benutzer in Ihrem AWS-Konto zu erstellen. Mit einem Identitätsanbieter (IdP) können Sie Ihre Benutzeridentitäten außerhalb von verwalten AWS und diesen externen Benutzeridentitäten Berechtigungen für den Zugriff auf AWS Ressourcen in Ihrem Konto erteilen. Weitere Informationen zu Verbünde und Identitätsanbietern finden Sie unter Identitätsanbieter und Verbund.

Erstellen von Rollen für Verbundbenutzer (Konsole)

Die Verfahren zum Erstellen einer Rolle für Verbundbenutzer sind vom gewählten Drittanbieter abhängig:

- Informationen zu OpenID Connect (OIDC) finden Sie unter. Eine Rolle für den OpenID Connect-Verbund erstellen (Konsole)
- Weitere Informationen über SAML 2.0 finden Sie unter Eine Rolle für den SAML 2.0-Verbund erstellen (Konsole).

Erstellen von Rollen für den Zugriff durch Verbundbenutzer (AWS CLI)

Die Schritte zum Erstellen einer Rolle für die unterstützten Identitätsanbieter (OIDC oder SAML) aus der AWS CLI sind identisch. Der Unterschied liegt im Inhalt der Vertrauensrichtlinie, die Sie in den notwendigen Schritten erstellen. Beginnen Sie mit den im Abschnitt Voraussetzungen beschriebenen Schritten für den verwendeten Anbietertyp:

- Weitere Informationen über OIDC-Anbieter finden Sie unter Voraussetzungen für die Erstellung einer Rolle für OIDC.
- Weitere Informationen über SAML-Anbieter finden Sie unter Voraussetzungen für das Erstellen einer Rolle für SAML.

Das Erstellen einer Rolle aus dem AWS CLI umfasst mehrere Schritte. Wenn Sie die Konsole verwenden, um eine Rolle zu erstellen, werden viele der Schritte für Sie erledigt, aber mit der müssen AWS CLI Sie jeden Schritt explizit selbst ausführen. Sie müssen die Rolle erstellen und ihr dann eine Berechtigungsrichtlinie zuweisen. Optional können Sie auch die Berechtigungsgrenze für Ihre Rolle festlegen.

So erstellen Sie eine Rolle für den Identitätsverbund (AWS CLI)

1. Erstellen Sie eine Rolle: aws iam create-role
2. Fügen Sie der Rolle eine Berechtigungsrichtlinie hinzu: aws iam attach-role-policy

   or

   Erstellen Sie eine Inline-Berechtigungsrichtlinie für die Rolle: aws iam put-role-policy
3. (Optional) Fügen Sie der Rolle benutzerdefinierte Attribute durch Zuweisen von Tags hinzu: aws iam tag-role

Weitere Informationen finden Sie unter Verwaltung von Tags in IAM-Rollen (AWS CLI oder AWS API).

4. (Optional) Legen Sie die Berechtigungsgrenze für die Rolle fest: aws iam put-role-permissions-boundary

   Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die eine Rolle haben kann. Bei den Berechtigungsgrenzen handelt es sich um eine erweiterte AWS Funktion.

Das folgende Beispiel zeigt die ersten beiden und häufigsten Schritte zum Anlegen einer Identitätsanbieter-Rolle in einer einfachen Umgebung. Dieses Beispiel erlaubt jedem Benutzer im `123456789012`-Konto, die Rolle zu übernehmen und den `example_bucket`-Amazon S3 Bucket anzuzeigen. In diesem Beispiel wird auch davon ausgegangen, dass Sie den AWS CLI auf einem Computer unter Windows ausführen und den bereits AWS CLI mit Ihren Anmeldeinformationen konfiguriert haben. Weitere Informationen finden Sie unter Konfigurieren der AWS Command Line Interface.

Das folgende Beispiel einer Vertrauensrichtlinie ist für eine mobile Anwendung vorgesehen, wenn sich der Benutzer mit Amazon Cognito anmeldet. In diesem Beispiel steht *`us-east:12345678-ffff-ffff-ffff -123456`* für die von Amazon Cognito zugewiesene Identitätspool-ID.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

Die folgende Berechtigungsrichtlinie erlaubt jedem, der die Rolle übernimmt, nur die Aktion `ListBucket` auf dem `example_bucket`-Amazon S3-Bucket durchzuführen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Um diese `Test-Cognito-Role`-Rolle zu erstellen, müssen Sie zuvor die vorhergehende Vertrauensrichtlinie mit dem Namen `trustpolicyforcognitofederation.json` und die vorherige Berechtigungsrichtlinie mit dem Namen `permspolicyforcognitofederation.json` im `policies`-Ordner Ihres lokalen `C:`-Laufwerks speichern. Sie können dann die folgenden Befehle verwenden, um die Rolle zu erstellen und die eingebundene Richtlinie hinzuzufügen.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json

# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Eine Rolle für den Verbundzugriff (API) erstellen AWS

Die Schritte zum Erstellen einer Rolle für die unterstützten Identitätsanbieter (OIDC oder SAML) aus der AWS CLI sind identisch. Der Unterschied liegt im Inhalt der Vertrauensrichtlinie, die Sie in den notwendigen Schritten erstellen. Beginnen Sie mit den im Abschnitt Voraussetzungen beschriebenen Schritten für den verwendeten Anbietertyp:

- Weitere Informationen über OIDC-Anbieter finden Sie unter Voraussetzungen für die Erstellung einer Rolle für OIDC.
- Weitere Informationen über SAML-Anbieter finden Sie unter Voraussetzungen für das Erstellen einer Rolle für SAML.

Um eine Rolle für den Identitätsverbund (AWS API) zu erstellen

1. Eine Rolle erstellen: CreateRole
2. Fügen Sie der Rolle eine Berechtigungsrichtlinie hinzu: AttachRolePolicy

   or

Erstellen Sie eine Inline-Berechtigungsrichtlinie für die Rolle: PutRolePolicy

3. (Optional) Fügen Sie dem Benutzer benutzerdefinierte Attribute hinzu, indem Sie Tags anhängen: TagRole

   Weitere Informationen finden Sie unter Verwaltung von Tags für IAM-Benutzer (AWS CLI oder AWS API).

4. (Optional) Legen Sie die Berechtigungsgrenze für die Rolle fest: PutRolePermissionsBoundary

   Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die eine Rolle haben kann. Bei den Berechtigungsgrenzen handelt es sich um eine erweiterte AWS Funktion.

Eine Rolle für den OpenID Connect-Verbund erstellen (Konsole)

Sie können die föderierten Identitätsanbieter von OpenID Connect (OIDC) verwenden, anstatt Benutzer in Ihrem zu erstellen AWS Identity and Access Management . AWS-Konto Mit einem Identitätsanbieter (IdP) können Sie Ihre Benutzeridentitäten außerhalb von verwalten AWS und diesen externen Benutzeridentitäten Berechtigungen für den Zugriff auf AWS Ressourcen in Ihrem Konto erteilen. Weitere Informationen zu Federation und IdPs finden Sie unter. Identitätsanbieter und Verbund

Voraussetzungen für die Erstellung einer Rolle für OIDC

Bevor Sie eine Rolle für den OIDC-Verbund erstellen können, müssen Sie zunächst die folgenden Voraussetzungen erfüllen.

Um sich auf die Erstellung einer Rolle für den OIDC-Verbund vorzubereiten

1. Melden Sie sich bei einem oder mehreren Services an, die eine Verbund-OIDC-Identität anbieten. Wenn Sie eine App erstellen, die Zugriff auf Ihre AWS Ressourcen benötigt, konfigurieren Sie Ihre App auch mit den Anbieterinformationen. Wenn Sie dies tun, erhalten Sie vom Anbieter eine Anwendungs- oder Zielgruppen-ID, die für Ihre Anwendung eindeutig ist. (Unterschiedliche Anbieter verwenden unterschiedliche Terminologie für diesen Prozess. In diesem Handbuch wird der Begriff configure, um Ihre App beim Anbieter zu identifizieren.) Sie können mehrere Anwendungen bei jedem Anbieter oder mehrere Anbieter mit einer einzigen Anwendung konfigurieren. Informationen zur Verwendung der Identitätsanbieter finden Sie im Folgendem:

   - Login with Amazon-Entwicklerzentrum

- Add Facebook Login to Your App or Website auf der Facebook-Entwickler-Website.
- Using OAuth 2.0 for Login (OpenID Connect) auf der Google-Entwickler-Website.

2. Nachdem Sie die erforderlichen Informationen vom Identitätsanbieter erhalten haben, erstellen Sie einen Identitätsanbieter in IAM. Weitere Informationen finden Sie unter Erstellen Sie einen OpenID Connect (OIDC) -Identitätsanbieter in IAM.

   > **Important**
   >
   > Wenn Sie einen OIDC-Identitätsanbieter von Google, Facebook oder Amazon Cognito verwenden, erstellen Sie keinen separaten IAM-Identitätsanbieter in der AWS Management Console. Diese OIDC-Identitätsanbieter sind bereits integriert AWS und stehen Ihnen zur Nutzung zur Verfügung. Überspringen Sie diesen Schritt, und wechseln Sie im folgenden Schritt zum Erstellen neuer Rollen mit Ihrem Identitätsanbieter.

3. Bereiten Sie die Richtlinien für die Rolle vor, die von den über einen Identitätsanbieter authentifizierten Benutzern übernommen werden sollen. Wie bei jeder Rolle enthält eine Rolle für eine mobile App zwei Richtlinien. Eine ist die Vertrauensrichtlinie, die angibt, wer die Rolle übernehmen kann. Die andere ist die Berechtigungsrichtlinie, die die AWS -Aktionen und -Ressourcen angibt, auf die die mobile App Zugriff hat oder nicht.

   Für das Internet empfehlen wir IdPs, Amazon Cognito zur Verwaltung von Identitäten zu verwenden. In diesem Fall verwenden Sie eine Vertrauensrichtlinie, ähnlich wie in diesem Beispiel.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

Ersetzen Sie `us-east-2:12345678-abcd-abcd-abcd-123456` durch die Identitätspool-ID, die Amazon Cognito Ihnen zuweist.

Wenn Sie einen OIDC-IdP manuell konfigurieren, müssen Sie bei der Erstellung der Vertrauensrichtlinie drei Werte verwenden, die sicherstellen, dass nur Ihre App die Rolle übernehmen kann:

- Verwenden Sie für das Element `Action` die Aktion `sts:AssumeRoleWithWebIdentity`.
- Verwenden Sie für das Element `Principal` die Zeichenfolge `{"Federated":`*`providerUrl/providerArn`*`}`.
  - Bei einigen gängigen OIDCs IdPs ist das eine URL. *`providerUrl`* Die folgenden Beispiele enthalten Methoden zur Angabe des Prinzips für einige häufig verwendete Methoden: IdPs

    ```
    "Principal":{"Federated":"cognito-identity.amazonaws.com"}
    ```

    ```
    "Principal":{"Federated":"www.amazon.com"}
    ```

    ```
    "Principal":{"Federated":"graph.facebook.com"}
    ```

    ```
    "Principal":{"Federated":"accounts.google.com"}
    ```
  - Verwenden Sie für andere OIDC-Anbieter den Amazon-Ressourcenname (ARN) des OIDC-Identitätsanbieters, den Sie in Step 2 erstellt haben, wie im folgenden Beispiel:

    ```
    "Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
    server.example.com"}
    ```
- Verwenden Sie für das Element `Condition` eine Bedingung `StringEquals`, um die Berechtigungen einzuschränken. Testen Sie die Identitätspool-ID für Amazon Cognito) oder die App-ID für andere Anbieter. Die Identitätspool-ID sollte mit der App-ID übereinstimmen, die Sie bei der Konfiguration der App mit dem Identitätsanbieter erhalten haben. Diese Übereinstimmung zwischen den IDs stellt sicher, dass die Anforderung von Ihrer App stammt.

> **Note**
>
> IAM-Rollen für Amazon Cognito Cognito-Identitätspools vertrauen darauf, dass der Service Principal `cognito-identity.amazonaws.com` die Rolle übernimmt. Rollen dieses Typs müssen mindestens einen Bedingungsschlüssel enthalten, um die Anzahl der Prinzipale einzuschränken, die die Rolle übernehmen können.

Zusätzliche Überlegungen gelten für Amazon Cognito Cognito-Identitätspools, die kontoübergreifende IAM-Rollen übernehmen. Die Vertrauensrichtlinien dieser Rollen müssen den `cognito-identity.amazonaws.com` Service Principal akzeptieren und den `aud` Bedingungsschlüssel enthalten, um die Rollenübernahme auf Benutzer aus Ihren vorgesehenen Identitätspools zu beschränken. Eine Richtlinie, die Amazon Cognito Cognito-Identitätspools ohne diese Bedingung vertraut, birgt das Risiko, dass ein Benutzer aus einem unbeabsichtigten Identitätspool die Rolle übernehmen kann. Weitere Informationen finden Sie unter Vertrauensrichtlinien für IAM-Rollen in der Standardauthentifizierung (Classic) im Amazon Cognito Developer Guide.

Erstellen Sie ein Bedingungselement ähnlich einem der folgenden Beispiele, je nach dem von Ihnen verwendeten Identitätsanbieter:

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

Verwenden Sie für OIDC-Anbieter die vollqualifizierte URL des OIDC-Identitätsanbieters mit dem Kontextschlüssel `aud` wie im folgenden Beispiel:

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

Note

Die Werte für den Prinzipal in der Vertrauensrichtlinie für die Rolle sind spezifisch für einen Identitätsanbieter. Eine Rolle für OIDC kann nur einen Prinzipal angeben. Wenn sich Benutzer in einer mobilen App über mehrere Identitätsanbieter anmelden können, müssen Sie daher für jeden Identitätsanbieter den Sie unterstützen möchten,

eine separate Rolle erstellen. Erstellen Sie separate Vertrauensrichtlinien für jeden Identitätsanbieter.

Wenn sich ein Benutzer über eine mobile App bei Login with Amazon anmeldet, würde das folgende Beispiel einer Vertrauensrichtlinie gelten. Im Beispiel steht *amzn1.application-oa2-123456* für die App-ID, die Amazon zuweist, wenn Sie die App mit Login with Amazon konfigurieren.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Wenn ein Benutzer eine mobile App verwendet, um sich von Facebook aus anzumelden, würde das folgende Beispiel einer Vertrauensrichtlinie gelten. In diesem Beispiel steht *111222333444555* für die App-ID, die Facebook zuweist.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Wenn ein Benutzer eine mobile App verwendet, um sich von Google aus anzumelden, würde das folgende Beispiel einer Vertrauensrichtlinie gelten. In diesem Beispiel steht *666777888999000* für die App-ID, die Google zuweist.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Wenn ein Benutzer eine mobile App verwendet, um sich von Amazon Cognito aus anzumelden, würde das folgende Beispiel einer Vertrauensrichtlinie gelten. In diesem Beispiel steht *us-east:12345678- ffff-ffff-ffff -123456* für die Identitätspool-ID, die Amazon Cognito zuweist.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Eine Rolle für OIDC erstellen

Nach dem Erstellen der vorbereitenden Schritte können Sie nun die Rolle in IAM erstellen. Das folgende Verfahren beschreibt, wie Sie die Rolle für den OIDC-Verbund in der erstellen. AWS

Management Console Informationen zum Erstellen einer Rolle über die AWS CLI AWS OR-API finden Sie unter. Erstellen von Rollen für externe Identitätsanbieter (Verbund)

> **Important**
>
> Wenn Sie Amazon Cognito verwenden, verwenden Sie die Amazon-Cognito-Konsole, um die Rollen einzurichten. Verwenden Sie andernfalls die IAM-Konsole, um eine Rolle für den OIDC-Verbund zu erstellen.

Um eine IAM-Rolle für den OIDC-Verbund 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. Wählen Sie im Navigationsbereich Rollen und dann Rolle erstellen.
3. Wählen Sie den OIDC-Rollentyp.
4. Wählen Sie für Identity provider (Identitätsanbieter), den Identitätsanbieter für Ihre Rolle:

   - Wenn Sie eine Rolle für einen einzelnen Web-Identitätsanbieter erstellen, wählen Sie Login with Amazon, Facebook oder Google.

     > **Note**
     >
     > Sie müssen für jeden Identitätsanbieter, den Sie unterstützen möchten, eine eigene Rolle erstellen.

   - Wenn Sie eine erweiterte Szenario-Rolle für Amazon Cognito erstellen möchten, wählen Sie Amazon Cognito.

     > **Note**
     >
     > Sie müssen eine Rolle manuell erstellen, um sie mit Amazon Cognito zu verwenden, wenn Sie an einem erweiterten Szenario arbeiten. Andernfalls kann Amazon Cognito Rollen für Sie erstellen. Weitere Informationen zu Amazon Cognito finden Sie unter Identitätspools (verbundene Identitäten) im Amazon-Cognito-Entwicklerhandbuch.

- Wenn Sie eine Rolle für GitHub Aktionen erstellen möchten, müssen Sie zunächst den GitHub OIDC-Anbieter zu IAM hinzufügen. Nachdem Sie den GitHub OIDC-Anbieter zu IAM hinzugefügt haben, wählen Sie token.actions.githubusercontent.com.

> **Note**
>
> Informationen zur Konfiguration, um dem OIDC als föderierte Identität AWS zu vertrauen GitHub, finden Sie unter GitHub Dokumente — Konfiguration von OpenID Connect in Amazon Web Services. Informationen zu bewährten Methoden zur Beschränkung des Zugriffs für Rollen, die mit dem IAM-IdP für verknüpft sind GitHub, finden Sie Konfiguration einer Rolle für den GitHub OIDC-Identitätsanbieter auf dieser Seite.

5. Geben Sie die Kennung für Ihre Anwendung ein. Die Marierung der Kennung ist abhängig von dem von Ihnen gewählten Anbieter:

- Wenn Sie eine Rolle für Login with Amazon erstellen, geben Sie die App-ID in das Feld Application ID (Anwendungs-ID) ein.
- Wenn Sie eine Rolle für Facebook erstellen, geben Sie die App-ID in das Feld Application ID (Anwendungs-ID) ein.
- Wenn Sie eine Rolle für Google erstellen, geben Sie die App-ID in das Feld Audience (Zielgruppe) ein.
- Wenn Sie eine Rolle für Amazon Cognito erstellen, geben Sie die ID des Identitäten-Pools, den Sie für Ihre Amazon-Cognito-Anwendungen erstellt haben, im Feld Identity Pool ID (Identitäten-Pool-ID) ein.
- Wenn Sie eine Rolle für GitHub Actions erstellen möchten, geben Sie die folgenden Details ein:
  - Wählen Sie für Audience (Zielgruppe) `sts.amazonaws.com`.
  - Geben Sie für GitHub Organisation den Namen Ihrer GitHub Organisation ein. Der Name der GitHub Organisation ist erforderlich und muss alphanumerisch sein und Bindestriche (-) enthalten. Sie können keine Platzhalterzeichen (* und?) verwenden im Namen der GitHub Organisation.
  - (Optional) Geben Sie für GitHub Repository den GitHub Repository-Namen ein. Wenn Sie keinen Wert angeben, wird standardmäßig ein Platzhalter (*) verwendet.

- (Optional) Geben Sie für GitHub Branch den GitHub Branch-Namen ein. Wenn Sie keinen Wert angeben, wird standardmäßig ein Platzhalter (*) verwendet.

6. (Optional:) Wählen Sie unter Bedingung (optional) die Option Bedingung hinzufügen aus, um weitere Bedingungen zu erstellen, die erfüllt werden müssen, damit Benutzer Ihrer Anwendung die durch die Rolle erteilten Berechtigungen verwenden können. Sie können beispielsweise eine Bedingung hinzufügen, die den Zugriff auf AWS Ressourcen nur für eine bestimmte IAM-Benutzer-ID gewährt. Sie können Bedingungen auch der Vertrauensrichtlinie hinzufügen, nachdem die Rolle erstellt wurde. Weitere Informationen finden Sie unter Ändern einer Rollenvertrauensrichtlinie (Konsole).
7. Überprüfen Sie Ihre OIDC-Informationen und wählen Sie dann Weiter.
8. IAM enthält eine Liste der AWS verwalteten und kundenverwalteten Richtlinien in Ihrem Konto. Wählen Sie die Richtlinie aus, die für die Berechtigungsrichtlinie verwendet werden soll, oder wählen Sie Create policy (Richtlinie erstellen), um eine neue Registerkarte im Browser zu öffnen und eine neue Richtlinie von Grund auf zu erstellen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien. Nachdem Sie die Richtlinie erstellt haben, schließen Sie die Registerkarte und kehren zur ursprünglichen Registerkarte zurück. Aktivieren Sie das Kontrollkästchen neben den Berechtigungsrichtlinien, über die OIDC-Benutzer verfügen sollen. Wenn Sie es vorziehen, zu diesem Zeitpunkt keine Richtlinien auszuwählen, können Sie sie der Rolle später hinzufügen. Standardmäßig hat eine Rolle keine Berechtigungen.
9. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature.

   Öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Sie Use a permissions boundary to control the maximum role permissions (Eine Berechtigungsgrenze verwenden, um die maximalen Rollen-Berechtigungen zu steuern). Wählen Sie die Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll.
10. Wählen Sie Weiter aus.
11. Geben Sie für Role name (Rollenname) einen Rollennamen ein. Rollennamen müssen innerhalb Ihres Unternehmens eindeutig sein. AWS-Konto Dabei wird zwischen Groß- und Kleinschreibung nicht unterschieden. Zum Beispiel können Sie keine Rollen erstellen, die sowohl **`PRODROLE`** als auch **`prodrole`** heißen. Da andere AWS Ressourcen möglicherweise auf die Rolle verweisen, können Sie den Namen der Rolle nicht bearbeiten, nachdem Sie sie erstellt haben.
12. (Optional) Geben Sie im Feld Description (Beschreibung) eine Beschreibung für die neue Rolle ein.
13. Um die Anwendungsfälle und Berechtigungen für die Rolle zu bearbeiten, wählen Sie in den Abschnitten Step 1: Select trusted entities (Schritt 1: Vertrauenswürdige Entitäten

auswählen) oder Step 2: Add permissions (Schritt 2: Berechtigungen hinzufügen) die Option Edit (Bearbeiten).

14. (Optional) Wenn Sie der Rolle Metadaten hinzufügen möchten, fügen Sie Tags als Schlüssel-Wert-Paare an. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
15. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Konfiguration einer Rolle für den GitHub OIDC-Identitätsanbieter

Wenn Sie GitHub als OpenID Connect (OIDC) Identity Provider (IdP) verwenden, empfiehlt es sich, die Entitäten einzuschränken, die die dem IAM-IdP zugeordnete Rolle übernehmen können. Wenn Sie eine Bedingungserklärung in die Vertrauensrichtlinie aufnehmen, können Sie die Rolle auf eine bestimmte GitHub Organisation, ein Repository oder eine Branche beschränken. Sie können den Bedingungsschlüssel `token.actions.githubusercontent.com:sub` mit Bedingungsoperatoren für Zeichenfolgen verwenden, um den Zugriff einzuschränken. Wir empfehlen, die Bedingung auf eine bestimmte Gruppe von Repositorys oder Zweigen innerhalb Ihrer GitHub Organisation zu beschränken. Informationen zur Konfiguration, um dem OIDC als föderierte Identität AWS zu vertrauen GitHub, finden Sie unter GitHub Dokumente — Konfiguration von OpenID Connect in Amazon Web Services.

Wenn Sie GitHub Umgebungen in Aktionsworkflows oder in OIDC-Richtlinien verwenden, empfehlen wir dringend, der Umgebung Schutzregeln hinzuzufügen, um zusätzliche Sicherheit zu gewährleisten. Verwenden Sie Deployment-Branches und -Tags, um einzuschränken, welche Branches und Tags in der Umgebung bereitgestellt werden können. Weitere Informationen zur Konfiguration von Umgebungen mit Schutzregeln finden Sie unter Deployment-Zweige und -Tags im GitHub Artikel Umgebungen für die Bereitstellung verwenden.

Wenn GitHub der OIDC-IdP der vertrauenswürdige Principal für Ihre Rolle ist, überprüft IAM die Bedingung der Rollenvertrauensrichtlinie, um sicherzustellen, dass der Bedingungsschlüssel vorhanden `token.actions.githubusercontent.com:sub` ist und dass sein Wert nicht nur ein Platzhalterzeichen (* und?) ist oder null. IAM führt diese Prüfung durch, wenn die Vertrauensrichtlinie erstellt oder aktualisiert wird. Wenn der Bedingungsschlüssel `token.actions.githubusercontent.com:sub` nicht vorhanden ist oder der Schlüsselwert die genannten Wertkriterien nicht erfüllt, schlägt die Anforderung fehl und gibt einen Fehler zurück.

> **Important**
>
> Wenn Sie den Bedingungsschlüssel nicht `token.actions.githubusercontent.com:sub` auf eine bestimmte Organisation oder ein bestimmtes Repository beschränken, können GitHub Aktionen von Organisationen oder Repositorys, auf die Sie keinen Einfluss haben, Rollen übernehmen, die dem GitHub IAM-IdP in Ihrem Konto zugeordnet sind. AWS

Das folgende Beispiel für eine Vertrauensrichtlinie beschränkt den Zugriff auf die definierte GitHub Organisation, das Repository und die Filiale. Der `token.actions.githubusercontent.com:sub` Wert des Bedingungsschlüssels im folgenden Beispiel ist das Standardformat für den Betreffwert, das von dokumentiert wird GitHub.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

Die folgende Beispielbedingung beschränkt den Zugriff auf die definierte GitHub Organisation und das definierte Repository, gewährt jedoch Zugriff auf alle Zweige innerhalb des Repositorys.

```
"Condition": {
  "StringEquals": {
```

```
            "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
    },
    "StringLike": {
      "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
    }
}
```

Die folgende Beispielbedingung beschränkt den Zugriff auf jedes Repository oder jede Filiale innerhalb der definierten GitHub Organisation. Wir empfehlen, dass Sie den Bedingungsschlüssel `token.actions.githubusercontent.com:sub` auf einen bestimmten Wert beschränken, der den Zugriff auf GitHub Aktionen innerhalb Ihrer GitHub Organisation einschränkt.

```
"Condition": {
    "StringEquals": {
            "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
    },
    "StringLike": {
      "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
    }
}
```

Weitere Informationen zu den OIDC-Verbundschlüsseln, die für Bedingungsprüfungen in Richtlinien verfügbar sind, finden Sie unter. Verfügbare Schlüssel für AWS den OIDC-Verbund

Eine Rolle für den SAML 2.0-Verbund erstellen (Konsole)

Sie können den SAML 2.0-Verbund verwenden, anstatt IAM-Benutzer in Ihrem zu erstellen. AWS-Konto Mit einem Identitätsanbieter (IdP) können Sie Ihre Benutzeridentitäten außerhalb von verwalten AWS und diesen externen Benutzeridentitäten Berechtigungen für den Zugriff auf AWS Ressourcen in Ihrem Konto erteilen. Weitere Informationen zu Verbünde und Identitätsanbietern finden Sie unter Identitätsanbieter und Verbund.

> **Note**
>
> Um die Verbundstabilität zu verbessern, empfehlen wir, dass Sie Ihren IdP und Ihren AWS -Verbund so konfigurieren, dass mehrere SAML-Anmeldeendpunkte unterstützt werden. Einzelheiten finden Sie im AWS Sicherheitsblogartikel How to use regional SAML endpoints for Failover.

Voraussetzungen für das Erstellen einer Rolle für SAML

Bevor Sie eine Rolle für den SAML-2.0-Verbund erstellen können, müssen Sie zunächst folgende Schritte ausführen.

So erstellen Sie eine Rolle für den SAML 2.0-Verbund

1. Bevor Sie eine Rolle für den SAML-basierten Verbund erstellen, müssen Sie einen SAML-Anbieter in IAM erstellen. Weitere Informationen finden Sie unter Erstellen Sie einen SAML-Identitätsanbieter in IAM.
2. Bereiten Sie die Richtlinien für die Rolle vor, die von den über SAML 2.0 authentifizierten Benutzern übernommen werden sollen. Wie bei jeder Rolle enthält eine Rolle für den SAML-Verbund zwei Richtlinien. Eine davon ist die Vertrauensrichtlinie, die angibt, wer die Rolle übernehmen kann. Die andere ist die IAM-Berechtigungsrichtlinie, die festlegt, auf welche AWS Aktionen und Ressourcen dem Verbundbenutzer Zugriff gewährt oder verweigert wird.

   Wenn Sie die Vertrauensrichtlinie für Ihre Rolle erstellen, müssen Sie drei Werte verwenden, um sicherzustellen, dass nur Ihre Anwendung die Rolle übernehmen kann:

   - Verwenden Sie für das Element `Action` die Aktion `sts:AssumeRoleWithSAML`.
   - Verwenden Sie für das Element `Principal` die Zeichenfolge `{"Federated":`*`ARNofIdentityProvider`*`}`. Ersetzen Sie *`ARNofIdentityProvider`* mit dem ARN des SAML-Identitätsanbieters, den Sie in Step 1 erstellt haben.
   - Verwenden Sie für das Element `Condition` eine `StringEquals`-Bedingung, um zu prüfen, ob das Attribut `saml:aud` aus der SAML-Antwort mit dem SAML-Verbund-Endpunkt für AWSübereinstimmt.

   Das folgende Beispiel ist auf eine Vertrauensrichtlinie für einen verbundenen SAML-Benutzer ausgelegt:

```
{
    "Version": "2012-10-17",
    "Statement": {
      "Effect": "Allow",
      "Action": "sts:AssumeRoleWithSAML",
      "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/PROVIDER-
NAME"},
      "Condition": {"StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/
saml"}}
```

```
    }
  }
```

Ersetzen Sie den Haupt-ARN durch den tatsächlichen ARN für den SAML-Anbieter, den Sie in IAM erstellt haben. Er enthält Ihre eigene Konto-ID und den Namen des Anbieters.

Erstellen einer Rolle für SAML

Nachdem Sie die erforderlichen Schritte durchgeführt haben, können Sie die Rolle für den SAML-basierten Verbund anlegen.

So erstellen Sie eine Rolle für den SAML-basierten Verbund

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 Roles (Rollen) und wählen Sie dann Create role (Rolle erstellen).
3. Wählen Sie den Rollentyp SAML 2.0 Federation.
4. Wählen Sie für Select a SAML provider (Einen SAML-Anbieter auswählen) den Anbieter für die Rolle.
5. Wählen Sie die Zugriffsebenen-Methode SAML 2.0.

   - Wählen Sie Nur programmatischen Zugriff zulassen aus, um eine Rolle zu erstellen, die programmgesteuert über die API oder übernommen werden kann. AWS AWS CLI
   - Wählen Sie Programmatischen AWS Management Console Zugriff zulassen aus, um eine Rolle zu erstellen, die programmgesteuert und von der aus übernommen werden kann. AWS Management Console

   Die erstellten Rollen ähneln einander, aber die Rolle, die auch über die Konsole übernommen werden kann, enthält eine Vertrauensrichtlinie mit einer bestimmten Bedingung. Diese Bedingung stellt ausdrücklich sicher, dass die SAML-Zielgruppe (`SAML:aud`-Attribut) auf den AWS -Anmelde-Endpunkt für SAML (https://signin.aws.amazon.com/saml) verweist.
6. Wenn Sie eine Rolle für den programmgesteuerten Zugriff erstellen, wählen Sie ein Attribut aus der Liste Attribute. Geben Sie dann im Feld Value (Wert) einen Wert ein, der in die Rolle aufgenommen werden soll. Dadurch wird der Rollenzugriff auf die Benutzer des Identitätsanbieters beschränkt, deren SAML-Authentifizierungsantwort (Zusicherung) das

angegebene Attribut enthält. Sie müssen mindestens ein Attribut angeben, um sicherzustellen, dass Ihre Rolle auf eine Benutzeruntergruppe in Ihrer Organisation beschränkt ist.

Wenn Sie eine Rolle für den programmgesteuerten Zugriff und den Zugriff über die Konsole erstellen, wird das Attribut `SAML:aud` automatisch hinzugefügt und auf die URL des AWS -SAML-Endpunkts (https://signin.aws.amazon.com/saml) gesetzt.

7. Um der Vertrauensrichtlinie weitere attributbezogene Bedingungen hinzuzufügen, wählen Sie Condition (optional) (Bedingung (optional)), wählen Sie die zusätzliche Bedingung aus und geben Sie einen Wert an.

   > **Note**
   >
   > In der Liste sind die am häufigsten verwendeten SAML-Attribute aufgeführt. IAM unterstützt weitere Attribute, die Sie zum Erstellen von Bedingungen verwenden können. (Eine Liste der unterstützten Attribute finden Sie unter Available Keys for SAML Federation.) Wenn Sie eine nicht in der Liste enthaltene Bedingung für ein unterstütztes SAML-Attribut benötigen, können Sie diese Bedingung manuell hinzufügen. Bearbeiten Sie dazu die Vertrauensrichtlinie, nachdem Sie die Rolle erstellt haben.

8. Überprüfen Sie Ihre SAML-2.0-Vertrauensinformationen und wählen Sie dann Next (Weiter).
9. IAM enthält eine Liste der AWS verwalteten und kundenverwalteten Richtlinien in Ihrem Konto. Wählen Sie die Richtlinie aus, die für die Berechtigungsrichtlinie verwendet werden soll, oder wählen Sie Create policy (Richtlinie erstellen), um eine neue Registerkarte im Browser zu öffnen und eine neue Richtlinie von Grund auf zu erstellen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien. Nachdem Sie die Richtlinie erstellt haben, schließen Sie die Registerkarte und kehren zur ursprünglichen Registerkarte zurück. Aktivieren Sie das Kontrollkästchen neben den Berechtigungsrichtlinien, über die OIDC-Verbundbenutzer verfügen sollen. Wenn Sie es vorziehen, zu diesem Zeitpunkt keine Richtlinien auszuwählen, können Sie sie der Rolle später hinzufügen. Standardmäßig hat eine Rolle keine Berechtigungen.
10. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature.

    Öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Sie Use a permissions boundary to control the maximum role permissions (Eine Berechtigungsgrenze verwenden, um die maximalen Rollen-Berechtigungen zu steuern). Wählen Sie die Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll.
11. Wählen Sie Weiter aus.
12. Wählen Sie Weiter: Prüfen aus.

13. Geben Sie für Role name (Rollennamen) einen Rollennamen ein. Rollennamen müssen innerhalb Ihres Unternehmens eindeutig sein. AWS-Konto Es wird hierbei nicht zwischen Groß- und Kleinschreibung unterschieden. z. B. können Sie keine Rollen erstellen, die **`PRODROLE`** bzw. **`prodrole`** heißen. Da andere AWS Ressourcen möglicherweise auf die Rolle verweisen, können Sie den Namen der Rolle nicht bearbeiten, nachdem sie erstellt wurde.
14. (Optional) Geben Sie im Feld Description (Beschreibung) eine Beschreibung für die neue Rolle ein.
15. Wählen Sie in den Abschnitten Step 1: Select trusted entities (Schritt 1: Vertrauenswürdige Entitäten auswählen) oder Step 2: Add permissions (Schritt 2: Berechtigungen hinzufügen) die Option Edit (Bearbeiten) aus, um die Anwendungsfälle und Berechtigungen für die Rolle zu bearbeiten.
16. (Optional) Fügen Sie der Rolle Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
17. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Nachdem Sie die Rolle erstellt haben, vervollständigen Sie schließlich die SAML-Vertrauensstellung, indem Sie die Software Ihres Identitätsanbieters mit Informationen zu AWS. Diese Informationen umfassen den Rollen, die Ihre Verbundbenutzer verwenden sollen. Dies wird als Konfiguration der Vertrauensstellung zwischen Ihrem Identitätsanbieter und AWS bezeichnet. Weitere Informationen finden Sie unter Konfigurieren Sie Ihren SAML 2.0-IdP mit dem Vertrauen der Vertrauensperson und dem Hinzufügen von Ansprüchen.

Erstellen einer Rolle mithilfe benutzerdefinierter Vertrauensrichtlinien (Konsole)

Sie können eine benutzerdefinierte Vertrauensrichtlinie erstellen, um den Zugriff zu delegieren und anderen die Ausführung von Aktionen in Ihrem AWS-Konto zu ermöglichen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien.

Weitere Informationen zur Verwendung von Rollen zum Delegieren von Berechtigungen finden Sie unter Rollenbegriffe und -konzepte.

Erstellen einer IAM-Rolle mithilfe benutzerdefinierter Vertrauensrichtlinien (Konsole)

Sie können die verwenden AWS Management Console , um eine Rolle zu erstellen, die ein IAM-Benutzer übernehmen kann. Gehen Sie beispielsweise davon aus, dass Ihr Unternehmen über mehrere Systeme verfügt AWS-Konten , um eine Entwicklungsumgebung von einer

Produktionsumgebung zu isolieren. Allgemeine Informationen zum Erstellen einer Rolle, mit der Benutzer im Entwicklungskonto auf Ressourcen in der Produktionsumgebung zugreifen können, finden Sie unter Beispielszenario mit getrennten Entwicklungs- und Produktionskonten.

Erstellen einer Rolle mithilfe einer benutzerdefinierten Vertrauensrichtlinie (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 Konsole auf Roles (Rollen) und wählen Sie dann Create role (Rolle erstellen).
3. Wählen Sie den Rollentyp Custom trust policy (Benutzerdefinierte Vertrauensrichtlinie).
4. Geben Sie im Abschnitt Custom trust policy (Benutzerdefinierte Vertrauensrichtlinie) die benutzerdefinierte Vertrauensrichtlinie für die Rolle ein, oder fügen Sie sie ein. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien.
5. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Next (Weiter).
6. Markieren Sie das Kontrollkästchen neben der benutzerdefinierten Vertrauensrichtlinie, die Sie vorher erstellt haben.
7. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature, das für Servicerollen verfügbar ist, aber nicht für servicegebundene Rollen.

   Öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Sie Use a permissions boundary to control the maximum role permissions (Eine Berechtigungsgrenze verwenden, um die maximalen Rollenberechtigungen zu steuern). IAM enthält eine Liste der AWS verwalteten und kundenverwalteten Richtlinien in Ihrem Konto. Wählen Sie die Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll.
8. Wählen Sie Weiter aus.
9. Für Role name (Rollenname) werden die Anpassungsmöglichkeiten für den Rollennamen durch den Service festgelegt. Wenn der Service den Namen der Rolle definiert, kann diese Option nicht bearbeitet werden. In anderen Fällen kann der Service ein Präfix für die Rolle festlegen und Ihnen das Eingeben eines optionalen Suffixes erlauben. Beim einigen Services können Sie den gesamten Namen Ihrer Rolle angeben.

   Geben Sie, wenn möglich, einen Rollennamen oder ein Rollennamen-Suffix ein. Rollennamen müssen innerhalb Ihres AWS-Konto Unternehmens eindeutig sein. Es wird hierbei nicht zwischen Groß- und Kleinschreibung unterschieden. z. B. können Sie keine Rollen erstellen, die

PRODROLE bzw. **prodrole** heißen. Da andere AWS Ressourcen möglicherweise auf die Rolle verweisen, können Sie den Namen der Rolle nicht bearbeiten, nachdem sie erstellt wurde.

10. (Optional) Geben Sie für Description (Beschreibung) eine Beschreibung für die neue Rolle ein.
11. Wählen Sie in den Abschnitten Step 1: Select trusted entities (Schritt 1: Vertrauenswürdige Entitäten auswählen) oder Step 2: Add permissions (Schritt 2: Berechtigungen hinzufügen) die Option Edit (Bearbeiten), um die benutzerdefinierte Richtlinie und Berechtigungen für die Rolle zu bearbeiten.
12. (Optional) Fügen Sie der Rolle Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
13. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Beispiele für Richtlinien zum Delegieren des Zugriffs

Die folgenden Beispiele zeigen, wie Sie einen AWS-Konto Zugriff auf die Ressourcen in einem anderen zulassen oder gewähren können AWS-Konto. Informationen zum Erstellen einer IAM-Richtlinie mithilfe dieser Beispiel-JSON-Richtliniendokumente finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

Themen

- Verwenden Sie Rollen, um den Zugriff auf die Ressourcen anderer AWS-Konto Ressourcen zu delegieren
- Verwenden einer Richtlinie zum Delegieren des Zugriffs auf Services
- Verwendung einer ressourcenbasierten Richtlinie zur Delegation des Zugriffs auf einen Amazon S3-Bucket in einem anderen Konto
- Verwendung einer ressourcenbasierten Richtlinie zur Delegation des Zugriffs auf eine Amazon SQS-Warteschlange in einem anderen Konto
- Delegieren des Zugriffs ist nicht möglich, wenn dem Konto der Zugriff verweigert wird

Verwenden Sie Rollen, um den Zugriff auf die Ressourcen anderer AWS-Konto Ressourcen zu delegieren

Ein Tutorial, in dem die Verwendung von IAM-Rollen zur Erteilung der Zugriffsberechtigung von Benutzern in einem Konto auf die AWS -Ressourcen in einem anderen Konto beschrieben wird, finden Sie unter Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen.

> **Important**
>
> Sie können den ARN für eine bestimmte Rolle oder einen bestimmten Benutzer in das `Principal`-Element einer rollenbasierten Vertrauensrichtlinie einschließen. Wenn Sie die Richtlinie speichern, AWS wandelt sie den ARN in eine eindeutige Prinzipal-ID um. Auf diese Weise wird das Risiko reduziert, dass jemand seine Berechtigungen durch Entfernen und Neuerstellen der Rolle oder des Benutzers erweitert. Normalerweise wird diese ID nicht in der Konsole angezeigt, da bei der Anzeige der Vertrauensrichtlinie auch eine Rückumwandlung zum ARN erfolgt. Wenn Sie jedoch die Rolle oder den Benutzer löschen, wird die Beziehung aufgehoben. Die Richtlinie wird nicht mehr angewendet, selbst wenn Sie den Benutzer oder die Rolle neu erstellen, da die Auftraggeber-ID der neuen Rolle nicht mit der in der Vertrauensrichtlinie gespeicherten ID übereinstimmt. In diesem Fall wird die Prinzipal-ID in der Konsole angezeigt, da sie nicht mehr einem ARN zugeordnet werden AWS kann. Das Ergebnis: Wenn Sie einen Benutzer oder eine Rolle löschen und neu erstellen, auf die beide im `Principal`-Element einer Vertrauensrichtlinie verwiesen wird, müssen Sie die Rolle bearbeiten, um den ARN zu ersetzen. Beim Speichern der Richtlinie wird dieser ARN in die neue Auftraggeber-ID umgewandelt.

Verwenden einer Richtlinie zum Delegieren des Zugriffs auf Services

Das folgende Beispiel zeigt eine Richtlinie, die einer Rolle angefügt werden kann. Die Richtlinie ermöglicht es zwei Diensten, Amazon EMR und AWS Data Pipeline, die Rolle zu übernehmen. Die Services können dann sämtliche Aufgaben ausführen, zu denen die Rolle infolge der zugewiesenen Berechtigungsrichtlinie berechtigt ist (nicht dargestellt). Geben Sie zur Angabe von mehreren Dienstauftraggeber nicht zwei `Service`-Elemente an. Sie können nur ein Element angeben. Verwenden Sie stattdessen ein Array mit mehreren Dienstauftraggeber als Wert eines einzelnen `Service`-Elements.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Verwendung einer ressourcenbasierten Richtlinie zur Delegation des Zugriffs auf einen Amazon S3-Bucket in einem anderen Konto

In diesem Beispiel verwendet Konto A eine ressourcenbasierte Richtlinie (eine Amazon S3-Bucket-Richtlinie), um dem Konto B vollen Zugriff auf den S3-Bucket in Konto A zu gewähren. Konto B erstellt dann eine IAM-Benutzerrichtlinie, um diesen Zugriff auf den Bucket in Konto A auf einen Benutzer im Konto B zu delegieren.

Die S3-Bucket-Richtlinie in Konto A sieht beispielsweise folgendermaßen. In diesem Beispiel wird der S3-Bucket im Konto A als mybucket bezeichnet und die Kontonummer von Konto B lautet 111122223333. In der Richtlinie werden keine einzelnen Benutzer oder Gruppen in Konto B angegeben, nur das Konto selbst.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

Alternativ kann Konto A Amazon S3-Zugriffskontrolllisten (ACLs) verwenden, um Konto B Zugriff auf einen S3-Bucket oder ein einzelnes Objekt innerhalb eines Buckets zu gewähren. In diesem Fall besteht der einzige Unterschied darin, wie Konto A dem Konto B die Zugriffsberechtigung erteilt. Konto B verwendet unverändert eine Richtlinie, um den Zugriff an eine IAM-Gruppe in Konto B zu delegieren, wie im nächsten Teil des Beispiels beschrieben. Weitere Informationen zur Zugriffkontrolle auf S3-Buckets und Objekte finden Sie unter Access Control im Benutzerhandbuch für Amazon Simple Storage Service.

Das Administrator von Konto B erstellt möglicherweise die folgende Beispielrichtlinie. Die Richtlinie gewährt Lesezugriff auf eine Gruppe oder einen Benutzer in Konto B. Die vorhergehende Richtlinie gewährt Zugriff auf Konto B. Einzelne Gruppen und Benutzer in Konto B haben jedoch erst Zugriff auf die Ressource, nachdem eine Gruppen- oder Benutzerrichtlinie explizit Berechtigungen für die Ressource gewährt hat. Die Berechtigungen in dieser Richtlinie können nur eine Untermenge der Berechtigungen in der vorhergehenden kontoübergreifenden Richtlinie sein. Konto B kann seinen Gruppen und Benutzern nicht mehr Berechtigungen erteilen, als jene, die das Konto A dem Konto B in der ersten Richtlinie erteilt hat. In dieser Richtlinie werden im Element `Action` explizit nur die Aktionen `List` zugelassen und das Element `Resource` in dieser Richtlinie stimmt mit der `Resource` in der von Konto A eingesetzten Bucketrichtlinie überein.

Zur Implementierung dieser Richtlinie verwendet Konto B IAM, um sie dem entsprechenden Benutzer (oder der entsprechenden Gruppe) in Konto B anzufügen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

Verwendung einer ressourcenbasierten Richtlinie zur Delegation des Zugriffs auf eine Amazon SQS-Warteschlange in einem anderen Konto

Im folgenden Beispiel verfügt Konto A über eine Amazon SQS-Warteschlange, die eine der Warteschlange angefügte ressourcenbasierte Richtlinie verwendet, um dem Konto B Zugriff auf die Warteschlange zu gewähren. Konto B verwendet dann eine IAM-Gruppenrichtlinie zum Delegieren des Zugriffs an eine Gruppe in Konto B.

Die folgende Beispielrichtlinie für eine Warteschlange erteilt Konto B die Berechtigung, `SendMessage`- und `ReceiveMessage`-Aktionen in der Warteschlange des Kontos A mit dem Namen queue1 ausschließlich am 30. November 2014 zwischen 12:00 und 15:00 Uhr auszuführen. Die Kontonummer des Kontos B lautet 1111-2222-3333. Konto A verwendet Amazon SQS zum Implementieren dieser Richtlinie.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

Die Richtlinie für das Delegieren des Zugriffs auf eine Gruppe in Konto B könnte folgendermaßen aussehen. Konto B verwendet IAM, um die Richtlinie einer Gruppe (oder einem Benutzer) anzufügen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

Im vorangegangenen Beispiel der IAM-Benutzerrichtlinie verwendet Konto B einen Platzhalter, um seinem Benutzer Zugriff auf alle Amazon SQS-Aktionen in der Warteschlange von Konto A zu gewähren. Konto B kann den Zugriff jedoch nur in dem Umfang delegieren, in dem Konto B Zugriff gewährt wurde. Die Gruppe in Konto B, die über die zweite Richtlinie verfügt, kann nur zwischen 12:00 Uhr und 15:00 Uhr am 30. November 2014 auf die Warteschlange zugreifen. Der Benutzer kann nur die Aktionen `SendMessage` und `ReceiveMessage` ausführen, wie in der Amazon SQS-Warteschlangenrichtlinie von Konto A definiert.

Delegieren des Zugriffs ist nicht möglich, wenn dem Konto der Zugriff verweigert wird

Ein AWS-Konto kann den Zugriff auf die Ressourcen eines anderen Kontos nicht delegieren, wenn das andere Konto den Zugriff auf das übergeordnete Konto des Benutzers ausdrücklich verweigert

hat. Die Zugriffsverweigerung wird den Benutzern in diesem Konto propagiert, unabhängig davon, ob diese Benutzer über Richtlinien verfügen, die ihnen Zugriffsberechtigungen erteilen.

In Konto A ist beispielsweise eine Bucketrichtlinie für den S3-Bucket in Konto A definiert, die dem Konto B den Zugriff auf den Bucket in Konto A explizit verweigert. Andererseits enthält Konto B eine IAM-Benutzerrichtlinie, die einem Benutzer in Konto B den Zugriff auf den Bucket in Konto A gewährt. Die explizite Zugriffsverweigerung auf den S3-Bucket in Konto A wird den Benutzern in Konto B propagiert. Sie überschreibt die IAM-Benutzerrichtlinie, die dem Benutzer in Konto B die Zugriffsberechtigung erteilt (weitere Informationen, wie Berechtigungen ausgewertet werden, finden Sie unter). Auswertungslogik für Richtlinien.)

Die Richtlinie des Buckets in Konto A sieht beispielsweise folgendermaßen. In diesem Beispiel wird der S3-Bucket im Konto A als mybucket bezeichnet und die Kontonummer von Konto B lautet 1111-2222-3333. Konto A verwendet Amazon S3; zum Implementieren dieser Richtlinie.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::mybucket/*"
  }
}
```

Diese explizite Zugriffsverweigerung überschreibt alle Richtlinien in Konto B, die zum Zugriff auf den S3-Bucket in Konto A berechtigen.

Verwenden von IAM-Rollen

Bevor ein Benutzer, Anwendungen oder Services eine von Ihnen erstellte Rolle verwenden können, müssen Sie Berechtigungen zum Wechseln zu dieser Rolle erteilen. Sie können jede Richtlinie verwenden, die Gruppen oder Benutzern zugewiesen ist, um die erforderlichen Berechtigungen zu gewähren. In diesem Abschnitt wird beschrieben, wie Sie Benutzern die Berechtigung zur Verwendung einer Rolle gewähren. Außerdem wird erklärt, wie der Benutzer über die Tools für Windows AWS Management Console PowerShell, die AWS Command Line Interface (AWS CLI) und die `AssumeRole`API zu einer Rolle wechseln kann.

> **Important**
>
> Wenn Sie die Rolle programmgesteuert anstatt in der IAM-Konsole erstellen, haben Sie die Möglichkeit, einen `Path` mit bis zu 512 Zeichen und einen `RoleName` mit bis zu 64 Zeichen hinzuzufügen. Wenn Sie jedoch beabsichtigen, eine Rolle mit der Funktion „Rolle wechseln“ in der zu verwenden AWS Management Console, dann die Kombination `Path` und `RoleName` darf 64 Zeichen nicht überschreiten.

Sie können die Rollen von der aus wechseln AWS Management Console. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Die verwendete Methode bestimmt, wer die Rolle annehmen und wie lange die Rollensitzung dauern kann. Bei der Verwendung von `AssumeRole*`-API-Vorgängen ist die von Ihnen angenommene IAM-Rolle die Ressource. Der Benutzer oder die Rolle, der/die `AssumeRole*`-API-Vorgänge aufruft, ist der Prinzipal.

Vergleichen von Methoden für die Nutzung von Rollen

Methode der Übernahme der Rolle	Wer die Rolle annehmen kann	Methode zum Festlegen der Lebensdauer der Anmeldeinformationen	Lebensdauer der Anmeldeinformationen (min | max | Standard)
AWS Management Console	Benutzer (durch Wechseln der Rollen)	Maximale Sitzungsdauer auf der Seite -Rollen-Zusammenfassung	15 Min. | Maximale Sitzungsdauer[2] | 1 Std.
`assume-role` -CLI- oder `AssumeRole` -API-Operation	Benutzer oder Rolle[1]	`duration-seconds` CLI- oder `DurationSeconds` API-Parameter	15 Min. | Maximale Sitzungsdauer[2] | 1 Std.

Methode der Übernahme der Rolle	Wer die Rolle annehmen kann	Methode zum Festlegen der Lebensdauer der Anmeldeinformationen	Lebensdauer der Anmeldeinformationen (min | max | Standard)
assume-role-with-saml -CLI- oder AssumeRoleWithSAML -API-Operation	Jeder Benutzer, der mit SAML authentifiziert wird	duration-seconds CLI- oder DurationSeconds API-Parameter	15 Min. | Maximale Sitzungsdauer[2] | 1 Std.
assume-role-with-web-identity -CLI- oder AssumeRoleWithWebIdentity -API-Operation	Jeder Benutzer, der über einen OIDC-Anbieter authentifiziert wurde	duration-seconds CLI- oder DurationSeconds API-Parameter	15 Min. | Maximale Sitzungsdauer[2] | 1 Std.
Konsolen-URL erstellt mit AssumeRole	Benutzer oder Rolle	SessionDuration HTML-Parameter in der URL	15 Min. | 12 Std. | 1 Std.
Konsolen-URL erstellt mit AssumeRoleWithSAML	Jeder Benutzer, der mit SAML authentifiziert wird	SessionDuration HTML-Parameter in der URL	15 Min. | 12 Std. | 1 Std.
Konsolen-URL erstellt mit AssumeRoleWithWebIdentity	Jeder Benutzer, der über einen OIDC-Anbieter authentifiziert wurde	SessionDuration HTML-Parameter in der URL	15 Min. | 12 Std. | 1 Std.

[1] Das Verwenden der Anmeldeinformationen für eine Rolle, um eine andere Rolle anzunehmen, wird als Verketten von Rollen bezeichnet. Wenn Sie die Verkettung von Rollen verwenden, sind Ihre neuen Anmeldeinformationen auf eine maximale Dauer von einer Stunde begrenzt. Wenn Sie Rollen verwenden, um Berechtigungen für Anwendungen zu erteilen, die auf EC2-Instances ausgeführt werden, unterliegen diese Anwendungen nicht dieser Einschränkung.

Diese Einstellung kann einen Wert zwischen 1 Stunde und 12 Stunden haben. Weitere Informationen zur maximalen Sitzungsdauer finden Sie unter Ändern einer Rolle. Diese Einstellung bestimmt die maximale Sitzungsdauer, die Sie anfordern können, wenn Sie die Anmeldeinformationen einer Rolle erhalten. Wenn Sie beispielsweise die API-Operationen AssumeRole* verwenden, um eine Rolle anzunehmen, können Sie mithilfe des Parameters eine Sitzungslänge angeben. `DurationSeconds` Verwenden Sie diesen Parameter, um die Länge der Rollensitzung von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle anzugeben. IAM-Benutzern wird die für die Rolle festgelegte maximale Sitzungsdauer oder die verbleibende Zeit in der Sitzung des Benutzers gewährt, je nachdem, welcher Wert geringer ist. Angenommen Sie, Sie eine maximale Dauer von 5 Stunden für eine Rolle festlegen. Ein IAM-Benutzer, der 10 Stunden lang bei der Konsole angemeldet wurde (ab dem Standardmaximum von 12), wechselt zur Rolle. Die verfügbare Rollensitzungsdauer beträgt 2 Stunden. Weitere Informationen zum Anzeigen des maximalen Werts für Ihre Rolle finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle weiter unten auf dieser Seite.

Hinweise

- Die Einstellung für die maximale Sitzungsdauer beschränkt keine Sitzungen, die von AWS -Services übernommen werden.
- Die Anmeldeinformationen für Amazon EC2 IAM-Rollen unterliegen nicht der in der Rolle konfigurierten maximalen Sitzungsdauer.
- Damit Benutzer die aktuelle Rolle innerhalb einer Rollensitzung wieder übernehmen können, geben Sie den Rollen-ARN oder AWS-Konto ARN als Principal in der Rollenvertrauensrichtlinie an. AWS-Services die Rechenressourcen wie Amazon EC2, Amazon ECS, Amazon EKS und Lambda bereitstellen, stellen temporäre Anmeldeinformationen bereit und aktualisieren diese Anmeldeinformationen automatisch. Dadurch wird sichergestellt, dass Sie immer über gültige Anmeldeinformationen verfügen. Für diese Dienste ist es nicht erforderlich, die aktuelle Rolle erneut anzunehmen, um temporäre Anmeldeinformationen zu erhalten. Wenn Sie jedoch beabsichtigen, Sitzungs-Tags oder eine Sitzungsrichtlinie zu übergeben, müssen Sie die aktuelle Rolle erneut annehmen. Informationen zum Ändern einer Rollenvertrauensrichtlinie zum

Hinzufügen der Hauptrollen ARN oder AWS-Konto ARN finden Sie unterÄndern einer Rollenvertrauensrichtlinie (Konsole).

Themen

- Anzeigen der maximalen Sitzungsdauer für eine Rolle
- Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen
- Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann
- Wechseln zu einer Rolle (Konsole)
- Wechseln zu einer IAM-Rolle (AWS CLI)
- Zu einer IAM-Rolle wechseln (Tools für Windows PowerShell)
- Zu einer IAM-Rolle (AWS API) wechseln
- Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden
- Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen

Anzeigen der maximalen Sitzungsdauer für eine Rolle

Sie können die maximale Sitzungsdauer für eine Rolle mithilfe der AWS Management Console AWS CLI oder AWS -API angeben. Wenn Sie eine API-Operation AWS CLI oder verwenden, um eine Rolle anzunehmen, können Sie einen Wert für den `DurationSeconds` Parameter angeben. Mit diesem Parameter können Sie die Dauer der Rollensitzung von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle festlegen. Bevor Sie den Parameter angeben, sollten Sie diese Einstellung für Ihre Rolle anzeigen. Wenn Sie einen Wert für den `DurationSeconds`-Parameter angeben, der höher als die maximale Einstellung ist, schlägt die Operation fehl.

So zeigen Sie die maximale Sitzungsdauer einer Rolle an (Konsole)

1. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus.
2. Wählen Sie den Namen der Rolle, die Sie anzeigen möchten.
3. Neben Maximale Sitzungsdauer sehen Sie die maximale Sitzungsdauer, die für die Rolle gewährt wird. Dies ist die maximale Sitzungsdauer, die Sie in Ihrem AWS CLI oder API-Vorgang angeben können.

So zeigen Sie die maximale Sitzungsdauer einer Rolle an (AWS CLI)

1. Wenn Sie den Namen der Rolle, die Sie übernehmen möchten, nicht kennen, führen Sie den folgenden Befehl aus, um die Rollen im Konto aufzulisten:

   - aws iam list-roles

2. Führen Sie zum Anzeigen der maximalen Sitzungsdauer einer Rolle folgenden Befehl. Anschließend zeigen Sie den Parameter für die maximale Sitzungsdauer an.

   - aws iam get-role

Um die Einstellung für die maximale Sitzungsdauer (AWS API) einer Rolle anzuzeigen

1. Wenn Sie den Namen der Rolle, die Sie übernehmen möchten, nicht kennen, rufen Sie die folgende Operation auf, um die Rollen im Konto aufzulisten:

   - ListRoles

2. Führen Sie zum Anzeigen der maximalen Sitzungsdauer der Rolle die folgende Operation. Anschließend zeigen Sie den Parameter für die maximale Sitzungsdauer an.

   - GetRole

Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen

Wenn ein Administrator eine Rolle für kontenübergreifenden Zugriff erstellt, richtet er eine Vertrauensstellung zwischen dem Konto, zu dem die Rolle gehört, sowie den Ressourcen (vertrauendes Konto) und dem Konto ein, das die Benutzer enthält (vertrauenswürdiges Konto). Zu diesem Zweck gibt der Administrator des vertrauenswürdigen Kontos die Nummer des vertrauenswürdigen Kontos als `Principal` in der Vertrauensrichtlinie der Rolle an. Dadurch kann jeder Benutzer im vertrauenswürdigen Konto potenziell die Rolle übernehmen. Zum Abschließen der Konfiguration muss der Administrator des vertrauenswürdigen Kontos bestimmten Gruppen oder Benutzern in diesem Konto die Berechtigung zum Wechseln zu der Rolle erteilen.

So erteilen Sie die Berechtigung zum Wechseln einer Rolle

1. Erstellen Sie als Administrator des vertrauenswürdigen Kontos eine neue Richtlinie für den Benutzer, oder bearbeiten Sie eine bestehende Richtlinie, um die erforderlichen Elemente hinzuzufügen. Details hierzu finden Sie unter Erstellen oder Bearbeiten der Richtlinie.

2. Wählen Sie dann, wie Sie die Rolleninformationen freigeben möchten:

   - Link zur Rolle: Sie können den Benutzern einen Link zusenden, mit dem sie zur Seite Switch Role (Rolle wechseln) gelangen, auf der sämtliche Angaben bereits ausgefüllt sind.
   - Konto-ID oder Alias: Geben Sie jedem Benutzer den Rollennamen sowie die Konto-ID-Nummer oder den Konto-Alias. Der Benutzer öffnet dann die Seite Switch Role (Rolle wechseln) und gibt die Details manuell ein.

   Details hierzu finden Sie unter Bereitstellen von Informationen an den Benutzer.

Beachten Sie, dass Sie die Rollen nur wechseln können, wenn Sie sich als IAM-Benutzer, als SAML-Verbundrolle oder mit einer Web-Identitäts-Verbundrolle anmelden. Sie können keine Rollen wechseln, wenn Sie als Root-Benutzer des AWS-Kontos angemeldet sind.

Important

Sie können die Rollen in nicht AWS Management Console zu einer Rolle wechseln, für die ein ExternalIdWert erforderlich ist. Sie können nur zu einer solchen Rolle wechseln, indem Sie die `AssumeRole`-API aufrufen, die den `ExternalId`-Parameter unterstützt.

Hinweise

- In diesem Thema werden die Richtlinien für einen Benutzer erörtert, da Sie letztendlich einem Benutzer die Berechtigungen zum Erledigen einer Aufgabe erteilen. Wir raten Ihnen jedoch davon ab, einem einzelnen Benutzer direkt Berechtigungen zu erteilen. Wenn ein Benutzer eine Rolle annimmt, werden ihm die mit dieser Rolle verknüpften Berechtigungen zugewiesen.
- Wenn Sie die Rollen in der wechseln AWS Management Console, verwendet die Konsole immer Ihre ursprünglichen Anmeldeinformationen, um den Switch zu autorisieren. Dies gilt unabhängig davon, ob Sie sich als IAM-Benutzer, als SAML-verbundene Rolle oder mit einer Web-Identität verbundenen Rolle anmelden. Wenn Sie beispielsweise zu RoleA wechseln, verwendet IAM; die Anmeldeinformationen Ihres ursprünglichen Benutzers oder Ihrer verbundenen Rolle, um festzustellen, ob Sie RoleA annehmen dürfen. Wenn Sie dann versuchen, zu RolleB zu wechseln, während Sie RolleA verwenden, werden Ihre ursprünglichen Benutzer- oder Verbundrollen-Anmeldeinformationen verwendet, um Ihren

Versuch zu autorisieren. Die Anmeldeinformationen für RoleA werden für diese Aktion nicht verwendet.

Themen

- Erstellen oder Bearbeiten der Richtlinie
- Bereitstellen von Informationen an den Benutzer

Erstellen oder Bearbeiten der Richtlinie

Eine Richtlinie, die dem Benutzer die Berechtigung zum Übernehmen einer Rolle gewährt, muss eine Anweisung enthalten, die für folgende Komponenten `Allow` gewährt:

- `sts:AssumeRole`-Aktion
- Amazon-Ressourcenname (ARN) der Rolle in einem `Resource`Element

Benutzer, die diese Richtlinie erhalten, dürfen die Rolle der aufgelisteten Ressource wechseln (entweder über eine Gruppenmitgliedschaft oder direkt angehängt).

> **Note**
>
> Wenn `Resource` auf * gesetzt ist, kann der Benutzer jede Rolle in jedem Konto übernehmen, das dem Konto des Benutzers vertraut. (Mit anderen Worten, die Vertrauensrichtlinie der Rolle gibt das Konto des Benutzers als `Principal` an). Als bewährte Methode empfehlen wir, dass Sie das Prinzip der Mindestberechtigung anwenden und den vollständigen ARN nur für die Rollen festlegen, die der Benutzer benötigt.

Das folgende Beispiel zeigt eine Richtlinie, mit der der Benutzer Rollen in nur einem Konto übernehmen kann. Zudem verwendet die Richtlinie einen Platzhalter (*), um anzugeben, dass der Benutzer nur zu einer Rolle wechseln kann, wenn der Rollenname mit den Buchstaben `Test` beginnt.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
```

```
    "Resource": "arn:aws:iam::account-id:role/Test*"
  }
}
```

> **Note**
>
> Die Berechtigungen, die die Rolle dem Benutzer gewährt, werden nicht zu den Berechtigungen addiert, die dem Benutzer bereits gewährt wurden. Wenn ein Benutzer zu einer Rolle wechselt, verzichtet er vorübergehend auf seine ursprünglichen Berechtigungen und erhält die Berechtigungen, die durch die Rolle gewährt werden. Wenn der Benutzer die Rolle beendet, werden die ursprünglichen Benutzerberechtigungen automatisch wiederhergestellt. Angenommen, die Berechtigungen des Benutzers erlauben die Arbeit mit Amazon EC2-Instaces, aber die Berechtigungsrichtlinie der Rolle gewährt diese Berechtigungen nicht. In diesem Fall kann der Benutzer, während er die Rolle verwendet, nicht mit Amazon EC2-Instances in der Konsole arbeiten. Darüber hinaus funktionieren über `AssumeRole` abgerufene temporäre Anmeldeinformationen nicht programmgesteuert mit Amazon EC2-Instances.

Bereitstellen von Informationen an den Benutzer

Nachdem Sie eine Rolle erstellt und dem Benutzer die Berechtigungen zum Wechseln zu dieser Rolle gewährt haben, müssen Sie dem Benutzer Folgendes bereitstellen:

- Der Name der Rolle
- Konto-ID oder Kontoalias mit der Rolle

Sie können den Vorgang für Ihre Benutzer erleichtern, indem Sie ihnen einen Link zusenden, der mit der Konto-ID und dem Rollennamen vorkonfiguriert ist. Sie können den Link zur Rolle sehen, nachdem Sie den Assistenten zum Erstellen einer Rolle abgeschlossen haben, indem Sie das Banner Rolle anzeigen auswählen oder auf der Seite Rollenübersicht für jede kontoübergreifende Rolle klicken.

Sie können auch das folgende Format verwenden, um den Link manuell zu erstellen. Ersetzen Sie die beiden Parameter in der Anforderung durch Ihre Konto-ID bzw. den Alias und den Rollennamen.

```
https://signin.aws.amazon.com/switchrole?
account=your_account_ID_or_alias&roleName=optional_path/role_name
```

Wir empfehlen Ihnen, Ihre Benutzer zum Thema Wechseln zu einer Rolle (Konsole) weiterzuleiten, um sie schrittweise durch den Prozess zu führen. Zur Behebung typischer Probleme beim Übernehmen einer Rolle vgl. Ich kann eine Rolle nicht übernehmen.

Überlegungen

- Wenn Sie die Rolle programmatisch erstellen, können Sie einen Pfad und einen Namen angeben. In diesem Fall müssen Sie Ihren Benutzern den vollständigen Pfad und den Rollennamen zur Eingabe auf der Seite Switch Role (Rolle wechseln) der AWS Management Console bereitstellen. Zum Beispiel: `division_abc/subdivision_efg/role_XYZ`.
- Wenn Sie die Rolle programmatisch erstellen, können Sie einen `Path` von bis zu 512 Zeichen und einen `RoleName` hinzufügen. Er kann eine Länge von bis zu 64 Zeichen umfassen. Um jedoch eine Rolle mit der Funktion „Rolle wechseln“ in der zu verwenden AWS Management Console, darf die Kombination `Path` 64 Zeichen `RoleName` nicht überschreiten.

- Aus Sicherheitsgründen können Sie in den AWS CloudTrail Protokollen nachlesen, wer eine Aktion in ausgeführt hat AWS. Sie können den `sts:SourceIdentity`-Bedingungsschlüssel in der Rollenvertrauensrichtlinie verwenden, damit Benutzer einen Sitzungsnamen angeben müssen, wenn sie eine Rolle übernehmen. Sie können beispielsweise verlangen, dass IAM-Benutzer ihren eigenen Benutzernamen als Sitzungsnamen angeben. Auf diese Weise können Sie feststellen, welcher Benutzer eine bestimmte Aktion in AWS ausgeführt hat. Weitere Informationen finden Sie unter `sts:SourceIdentity`. Sie können auch `sts:RoleSessionName` verwenden, um von den Benutzern zu verlangen, dass sie einen Sitzungsnamen angeben, wenn sie eine Rolle übernehmen. Auf diese Weise können Sie zwischen Rollensitzungen unterscheiden, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird.

Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Dies erlaubt dem Service, später die Rolle anzunehmen und in Ihrem Auftrag Aktionen durchzuführen. Für die meisten Services müssen Sie die Rolle nur einmal während der Einrichtung an den Service übergeben, und nicht jedes Mal, wenn der Service die Rolle annimmt. Angenommen, es wird eine Anwendung auf einer Amazon EC2-Instance ausgeführt. Diese Anwendung erfordert temporäre Anmeldeinformationen zur Authentifizierung und Berechtigungen für die Autorisierung der Anwendung, damit diese Aktionen in AWS durchführen kann. Wenn Sie die Anwendung einrichten,

müssen Sie eine Rolle an Amazon EC2 übergeben, die mit der Instance verwendet wird, die diese Anmeldeinformationen bereitstellt. Sie definieren die Berechtigungen für die Anwendungen, die in dieser Instance ausgeführt werden, indem Sie der Rolle eine IAM-Richtlinie anfügen. Die Anwendung nimmt die Rolle immer an, wenn die Aktionen im Rahmen der Rolle ausgeführt werden.

Um eine Rolle (und ihre Berechtigungen) an einen AWS Dienst zu übergeben, muss ein Benutzer über die erforderlichen Berechtigungen verfügen, um die Rolle an den Dienst weiterzugeben. So können Administratoren sicherstellen, dass nur berechtigte Benutzer einen Service mit einer Rolle konfigurieren können, über die Berechtigungen verliehen werden. Damit ein Benutzer eine Rolle an einen AWS Dienst übergeben kann, müssen Sie dem IAM-Benutzer, der Rolle oder der Gruppe des Benutzers die `PassRole` entsprechende Berechtigung erteilen.

⚠ Warning

- Sie können die `PassRole` Berechtigung nur verwenden, um eine IAM-Rolle an einen Dienst zu übergeben, der dasselbe AWS Konto verwendet. Um eine Rolle in Konto A an einen Service in Konto B zu übergeben, müssen Sie zunächst eine IAM-Rolle in Konto B erstellen, die die Rolle von Konto A übernehmen kann. Anschließend kann die Rolle in Konto B an den Service übergeben werden. Details hierzu finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.
- Versuchen Sie nicht zu kontrollieren, wer eine Rolle weitergeben kann, indem Sie die Rolle taggen und dann den `ResourceTag`-Bedingungsschlüssel in einer Richtlinie mit der `iam:PassRole`-Aktion verwenden. Dieser Ansatz führt nicht zu zuverlässigen Ergebnissen.

Wenn Sie die `PassRole` Berechtigung festlegen, sollten Sie sicherstellen, dass ein Benutzer keine Rolle weitergibt, bei der die Rolle über mehr Berechtigungen verfügt, als Sie dem Benutzer zuweisen möchten. Beispielsweise darf Alice möglicherweise keine Amazon S3 S3-Aktionen ausführen. Wenn Alice eine Rolle an einen Service übergeben könnte, der Amazon S3 S3-Aktionen zulässt, könnte der Service bei der Ausführung des Jobs Amazon S3 S3-Aktionen im Namen von Alice ausführen.

Wenn Sie eine serviceverknüpfte Rolle angeben, müssen Sie zudem über die Berechtigung verfügen, diese Rolle an den Service übergeben. Einige Services erstellen automatisch eine serviceverknüpfte Rolle in Ihrem Konto, wenn Sie eine Aktion in diesem Service durchführen. Zum Beispiel erstellt Amazon EC2 Auto Scaling die serviceverknüpfte Rolle `AWSServiceRoleForAutoScaling` für Sie, wenn Sie das erste Mal eine Auto-Scaling-Gruppe erstellen. Wenn Sie versuchen, die

Berechtigung anzugeben, wenn Sie eine Auto-Scaling-Gruppe erstellen, und Sie haben keine `iam:PassRole`-Berechtigung, erhalten Sie einen Fehler. Wenn Sie die Rolle nicht explizit angeben, ist die `iam:PassRole`-Berechtigung nicht erforderlich, und standardmäßig wird die `AWSServiceRoleForAutoScaling`-Rolle für alle Operationen verwendet, die für diese Gruppe ausgeführt werden. Informationen dazu, welche Services serviceverknüpfte Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Um zu erfahren, welche Services automatisch eine serviceverknüpfte Rolle erstellen, wenn Sie eine Aktion in diesem Service durchführen, wählen Sie den Link Yes (Ja) und zeigen Sie die serviceverknüpfte Rollendokumentation für den Service an.

Ein Benutzer kann einen Rollen-ARN als Parameter in einer API-Operation übergeben, die die Rolle für die Zuweisung von Berechtigungen für den Service verwendet. Der Service überprüft dann, ob der betreffende Benutzer über die Berechtigung `iam:PassRole` verfügt. Damit der Benutzer nur genehmigte Rollen übergeben kann, können Sie die Berechtigung `iam:PassRole` mit dem Element `Resources` der IAM-Richtlinienanweisung filtern.

Sie können das `Condition` Element in einer JSON-Richtlinie verwenden, um den Wert von Schlüsseln zu testen, die im Anforderungskontext aller AWS Anfragen enthalten sind. Weitere Informationen zur Verwendung von Bedingungsschlüsseln in einer Richtlinie finden Sie unter IAM-JSON-Richtlinienelemente: Condition. Der `iam:PassedToService`-Bedingungsschlüssel kann verwendet werden, um den Auftraggeber des Dienstes anzugeben, an den eine Rolle übergeben werden kann. Weitere Informationen zur Verwendung des `iam:PassedToService` Bedingungsschlüssels in einer Richtlinie finden Sie unter iam: PassedToService.

Beispiel 1

Angenommen, Sie möchten einem Benutzer die Berechtigung verleihen, beim Starten einer Instance beliebige genehmigte Rollen an den Amazon-EC2-Service zu übergeben. Dafür sind drei Elemente nötig:

- Eine IAM-Berechtigungsrichtlinie, die der Rolle zugeordnet ist und über die festgelegt wird, was mit der Rolle getan werden kann. Weisen Sie nur Berechtigungen für Aktionen zu, die für die Rolle erforderlich sind, und nur für die Ressourcen, die von der Rolle für diese Aktionen benötigt werden. Sie können eine AWS verwaltete oder vom Kunden erstellte IAM-Berechtigungsrichtlinie verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": {
```

```
        "Effect": "Allow",
        "Action": [ "A list of the permissions the role is allowed to use" ],
        "Resource": [ "A list of the resources the role is allowed to access" ]
    }
}
```

- Eine Vertrauensrichtlinie für die Rolle, die es dem Service ermöglicht, die Rolle anzunehmen. Sie können beispielsweise der Rolle mit der Aktion `UpdateAssumeRolePolicy` die folgende Vertrauensrichtlinie zuweisen. Über diese Vertrauensrichtlinie kann Amazon EC2 die Rolle und deren Berechtigungen verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
        "Effect": "Allow",
        "Principal": { "Service": "ec2.amazonaws.com" },
       "Action": "sts:AssumeRole"
    }
}
```

- Eine IAM-Berechtigungsrichtlinie, die dem IAM-Benutzer zugeordnet ist und diesem ermöglicht, nur genehmigte Rollen zu übergeben. Sie fügen normalerweise `iam:GetRole` zu `iam:PassRole`, damit der Benutzer die Details weitergeben kann. In diesem Beispiel kann der Benutzer nur Rollen weitergeben, die im angegebenen Konto mit Namen vorhanden sind, die mit `EC2-roles-for-XYZ-` beginnen:

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:GetRole",
            "iam:PassRole"
        ],
        "Resource": "arn:aws:iam::account-id:role/EC2-roles-for-XYZ-*"
    }]
}
```

Der Benutzer kann jetzt eine Amazon EC2-Instance mit einer zugewiesenen Rolle starten. Anwendungen auf der Instance können über die Metadaten des Instance-Profils auf temporäre Anmeldeinformationen für die Rolle zugreifen. Die der Rolle zugeordneten Berechtigungsrichtlinien legen fest, welche Aktionen die Instance ausführen kann.

Beispiel 2

Amazon Relational Database Service (Amazon RDS) unterstützt die Feature namens „Verbesserte Überwachung“. Mit dieses Feature kann Amazon RDS eine Datenbank-Instance mithilfe eines Agenten überwachen. Es ermöglicht Amazon RDS auch, Metriken in Amazon Logs zu CloudWatch protokollieren. Um dieses Feature zu aktivieren, müssen Sie eine Servicerolle erstellen, um Amazon RDS die Berechtigung zum Überwachen und Speichern von Metriken in Ihren Protokollen zu gewähren.

So erstellen Sie eine IAM-Rolle für Enhanced Monitoring in Amazon RDS

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 Roles (Rollen) und anschließend Create role (Rolle erstellen).
3. Wählen Sie den Rollentyp AWS Service und dann für Anwendungsfälle für andere AWS-Services den RDS-Dienst aus. Wählen Sie RDS – Enhanced Monitoring (RDS – erweiterte Überwachung) und Next (Weiter) aus.
4. Wählen Sie die EnhancedMonitoringRoleAmazonRDS-Berechtigungsrichtlinie aus.
5. Wählen Sie Weiter aus.
6. Geben Sie unter Role name (Rollenname) einen Rollennamen ein, der Ihnen hilft, den Zweck dieser Rolle zu identifizieren. Rollennamen müssen innerhalb Ihres AWS-Konto Unternehmens eindeutig sein. Wenn ein Rollenname in einer Richtlinie oder als Teil eines ARN verwendet wird, muss die Groß-/Kleinschreibung des Rollennamens beachtet werden. Wenn Kunden in der Konsole ein Rollenname angezeigt wird, beispielsweise während des Anmeldevorgangs, wird die Groß-/Kleinschreibung des Rollennamens nicht beachtet. Da verschiedene Entitäten möglicherweise auf die Rolle verweisen, können Sie den Namen der Rolle nach der Erstellung nicht mehr bearbeiten.
7. (Optional) Geben Sie unter Role description (Rollenbeschreibung) eine Beschreibung für die neue Rolle ein.
8. (Optional) Fügen Sie dem Benutzer Metadaten hinzu, indem Sie Markierungen als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.

9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Der Rolle wird automatisch eine Vertrauensrichtlinie zugewiesen, die dem Service `monitoring.rds.amazonaws.com` die Berechtigung zum Übernehmen der Rolle gewährt. Nun kann Amazon RDS; alle durch die Richtlinie `AmazonRDSEnhancedMonitoringRole` gewährten Aktionen ausführen.

Der Benutzer, der auf Enhanced Monitoring zugreifen soll, benötigt eine Richtlinie wie die folgende, die eine Anweisung enthält, die es dem Benutzer ermöglicht, die RDS-Rollen aufzulisten und eine Anweisung, die es dem Benutzer ermöglicht, die Rolle zu übergeben. Verwenden Sie Ihre Kontonummer und ersetzen Sie den Rollennamen durch den Namen, den Sie in Schritt 6 eingegeben haben.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
    "Effect": "Allow",
    "Action": [ "iam:PassRole" ],
    "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
}
```

Sie können diese Anweisung mit Anweisungen in einer anderen Richtlinie kombinieren oder eine eigene Richtlinie dafür erstellen. Um stattdessen festzulegen, dass der Benutzer jede Rolle übergeben kann, die mit `RDS-` beginnt, ersetzen Sie wie folgt den Rollennamen im Ressourcen-ARN durch einen Platzhalter.

```
    "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

`iam:PassRole`-Aktionen in AWS CloudTrail -Protokollen

`PassRole`ist kein API-Aufruf. `PassRole`ist eine Berechtigung, was bedeutet, dass keine CloudTrail Protokolle für IAM `PassRole` generiert werden. Um zu überprüfen, welche Rollen an welche AWS-Services übergeben wurden CloudTrail, müssen Sie das CloudTrail Protokoll überprüfen, das die

AWS Ressource, der die Rolle zugewiesen wurde, erstellt oder geändert hat. Beispielsweise wird eine Rolle bei ihrer Erstellung an eine AWS Lambda Funktion übergeben. Das Protokoll für die `CreateFunction`-Aktion zeigt eine Aufzeichnung der Rolle, die an die Funktion übergeben wurde.

Wechseln zu einer Rolle (Konsole)

Eine Rolle legt eine Gruppe von Berechtigungen fest, die Sie für den Zugriff auf AWS -Ressourcen, die Sie benötigen, verwenden können. In dieser Hinsicht ist sie mit einem IAM-Benutzer AWS Identity and Access Management vergleichbar. Wenn Sie sich als Benutzer anmelden, erhalten Sie einen bestimmten Satz von Berechtigungen. Sie melden sich nicht bei einer Rolle an, aber sobald Sie angemeldet sind, können Sie zu einer Rolle wechseln. Dadurch werden Ihre ursprünglichen Benutzerberechtigungen vorübergehend zurückgestellt und Sie erhalten stattdessen die der Rolle zugewiesenen Berechtigungen. Die Rolle kann sich in Ihrem eigenen Konto oder jedem anderen AWS-Konto befinden. Weitere Informationen zu Rollen, ihren Vorteilen sowie zu ihrer Erstellung finden Sie unter IAM-Rollen und Erstellen von IAM-Rollen.

> **Important**
>
> Die Berechtigungen Ihres -Benutzers und alle Rollen, zu denen Sie wechseln, können nicht kumuliert werden. Es ist nur jeweils ein Satz von Berechtigungen aktiv. Wenn Sie zu einer Rolle wechseln, geben Sie Ihre Benutzerberechtigungen temporär auf und arbeiten mit den Berechtigungen, die der Rolle zugeordnet sind. Wenn Sie die Rolle verlassen, werden Ihre Benutzerberechtigungen automatisch wiederhergestellt.

Wenn Sie in der die Rollen wechseln AWS Management Console, verwendet die Konsole immer Ihre ursprünglichen Anmeldeinformationen, um den Switch zu autorisieren. Dies gilt unabhängig davon, ob Sie sich als IAM-Benutzer, als Benutzer im IAM Identity Center, als SAML-verbundene Rolle oder mit einer Web-Identität verbundenen Rolle anmelden. Wenn Sie beispielsweise zu RoleA wechseln, verwendet IAM die Anmeldeinformationen Ihres ursprünglichen Benutzers oder Ihrer verbundenen Rolle, um festzustellen, ob Sie RoleA übernehmen dürfen. Wenn Sie dann zu RoleB wechseln, während Sie RoleA verwenden, verwendet AWS immer noch Ihre ursprünglichen Benutzer- oder Verbundrollenanmeldeinformationen, um den Wechsel zu autorisieren, nicht die Anmeldeinformationen für RoleA.

Wissenswerte Informationen zum Wechseln von Rollen in der Konsole

Dieser Abschnitt enthält weitere Informationen zur Verwendung der IAM-Konsole zum Wechseln zu einer Rolle.

Hinweise:

- Sie können die Rollen nicht wechseln, wenn Sie sich als anmelden. Root-Benutzer des AWS-Kontos Sie können die Rolle wechseln, wenn Sie sich als IAM-Benutzer, als Benutzer im IAM Identity Center, als SAML-verbundene Rolle oder als Web-Identitätsverbundene Rolle anmelden.
- Sie können die Rollen in nicht AWS Management Console zu einer Rolle wechseln, für die ein ExternalIdWert erforderlich ist. Sie können nur zu einer solchen Rolle wechseln, indem Sie die `AssumeRole`-API aufrufen, die den `ExternalId`-Parameter unterstützt.

- Wenn Sie von Ihrem Administrator einen Link erhalten, klicken Sie auf den Link und fahren Sie direkt mit Schritt Step 5 in der folgenden Anleitung fort. Über den Link gelangen Sie zur entsprechenden Webseite, in der bereits die Konto-ID (oder der Alias) sowie der Rollenname eingetragen sind.
- Sie können den Link manuell erstellen und dann den Schritt Step 5 in der folgenden Anleitung auslassen. Verwenden Sie beim Erstellen des Links das folgende Format:

  `https://signin.aws.amazon.com/switchrole?account=`*`account_id_number`*`&roleName=`*`role_name`*`&displayName=`*`text_to_display`*

  Ersetzen Sie die folgenden Platzhalter:

  - *`account_id_number`* – 12-stellige Konto-ID, die Sie vom Administrator erhalten. Alternativ kann der Administrator einen Kontoalias erstellen, sodass die URL den Kontonamen anstelle der Konto-ID enthält. Weitere Informationen finden Sie unter Benutzertypen im AWS-Anmeldung - Benutzerhandbuch.
  - *`role_name`* – Name der zu übernehmenden Rolle. Sie finden diesen Wert am Ende des ARN der Rolle. Geben Sie beispielsweise den Rollennamen `TestRole` als ARN der Rolle `arn:aws:iam::123456789012:role/TestRole` an.
  - (Optional) *`text_to_display`* – Geben Sie Text ein, der anstelle des Benutzernamens in der Navigationsleiste angezeigt werden soll, wenn die Rolle aktiv ist.
- Sie können die Rollen anhand der von Ihrem Administrator bereitgestellten Informationen manuell wechseln, indem Sie die folgenden Verfahren anwenden.

Wenn Sie die Rollen wechseln, dauert Ihre AWS Management Console Sitzung standardmäßig 1 Stunde. IAM-Benutzersitzungen sind standardmäßig 12 Stunden. IAM-Benutzer wird die für die Rolle festgelegte maximale Sitzungsdauer oder die verbleibende Zeit in der Sitzung des Benutzers gewährt, je nachdem, welcher Wert geringer ist. Nehmen Sie beispielsweise an, dass eine maximale Sitzungsdauer von 10 Stunden für eine Rolle festgelegt ist. Ein IAM-Benutzer wurde 8 Stunden lang bei der Konsole angemeldet, wenn er sich entscheidet, zur Rolle zu wechseln. In der Benutzersitzung sind 4 Stunden verbleibend, sodass die zulässige Rollensitzungsdauer 4 Stunden beträgt. Die folgende Tabelle zeigt, wie die Sitzungsdauer für einen IAM-Benutzer beim Wechseln von Rollen in der Konsole ermittelt wird.

IAM-Benutzer Konsolenrollensitzungsdauer

Die verbleibende IAM-Benutzersitzung ist...	Rollensitzungsdauer beträgt…
Maximale Sitzungsdauer unter Rolle	Verbleibende Zeit in Benutzersitzung
Maximale Sitzungsdauer größer als Rolle	Maximale Sitzungsdauer
Gleich der maximalen Sitzungsdauer der Rolle	Maximaler Wert für die Sitzungsdauer (ungefähr)

> **Note**
>
> Einige AWS Servicekonsolen können Ihre Rollensitzung automatisch verlängern, wenn sie abläuft, ohne dass Sie etwas unternehmen. Einige werden Sie möglicherweise aufgefordert, Ihre Browserseite neu zu laden, um Ihre Sitzung erneut zu authentifizieren.

Zur Behebung typischer Probleme beim Übernehmen einer Rolle vgl. Ich kann eine Rolle nicht übernehmen.

So wechseln Sie zu einer Rolle (Konsole)

1. Melden Sie sich AWS Management Console als IAM-Benutzer an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie in der IAM-Konsole rechts oben auf der Navigationsleiste den Benutzernamen. Dieser hat normalerweise folgendes Format: ***Benutzername*@*Konto_ID_Nummer_oder_Alias***.
3. Wählen Sie Switch Role. Wenn Sie diese Option erstmalig auswählen, wird eine Seite mit weiteren Informationen angezeigt. Lesen Sie sich diese Informationen durch und klicken Sie dann auf Switch Role (Rolle wechseln). Wenn Sie die Cookies Ihres Browsers löschen, kann die Seite erneut angezeigt werden.
4. Geben Sie auf der Seite Switch Role (Rolle wechseln) die Konto-ID-Nummer oder den Kontoalias sowie den Namen der Rolle ein, die Sie von Ihrem Administrator erhalten haben.

   > **Note**
   >
   > Wenn der Administrator die Rolle mit einem Pfad erstellt hat, z. B. `division_abc/subdivision_efg/roleToDoX`, müssen Sie den vollständigen Pfad sowie den Namen im Feld Role (Rolle) eingeben. Wenn Sie nur den Rollennamen eingeben oder der kombinierte `Path` und `RoleName` 64 Zeichen überschreiten, schlägt der Wechsel der Rolle fehl. Diese Begrenzung wird durch die Browser-Cookies vorgegeben, in denen der Rollenname gespeichert wird. Wenn dies der Fall ist, wenden Sie sich an Ihren Administrator und bitten Sie Ihn, die Größe des Pfads und Rollennamens zu verringern.

5. (Optional) Wählen Sie einen Anzeigename aus. Geben Sie den Text ein, der anstelle Ihres Benutzernamens in der Navigationsleiste erscheinen soll, wenn diese Rolle aktiv ist. Es wird ein Name basierend auf den Konto- und Rolleninformationen vorgeschlagen. Sie können jedoch einen beliebigen Namen wählen. Außerdem können Sie eine Farbe auswählen, in der der Anzeigename hervorgehoben wird. Der Name und die Farbe helfen Ihnen dabei zu erkennen, dass die Rolle aktiv ist und Sie dadurch möglicherweise über andere Berechtigungen verfügen. Sie können beispielsweise für eine Rolle, die Ihnen Zugriff auf die Testumgebung gewährt, den Display Name (Anzeigename) **`Test`** und die Color (Farbe) grün auswählen. Für die Rolle, die Ihnen Zugriff auf die Produktionsumgebung gewährt, können Sie den Display Name (Anzeigename) **`Production`** und die Color (Farbe) rot auswählen.

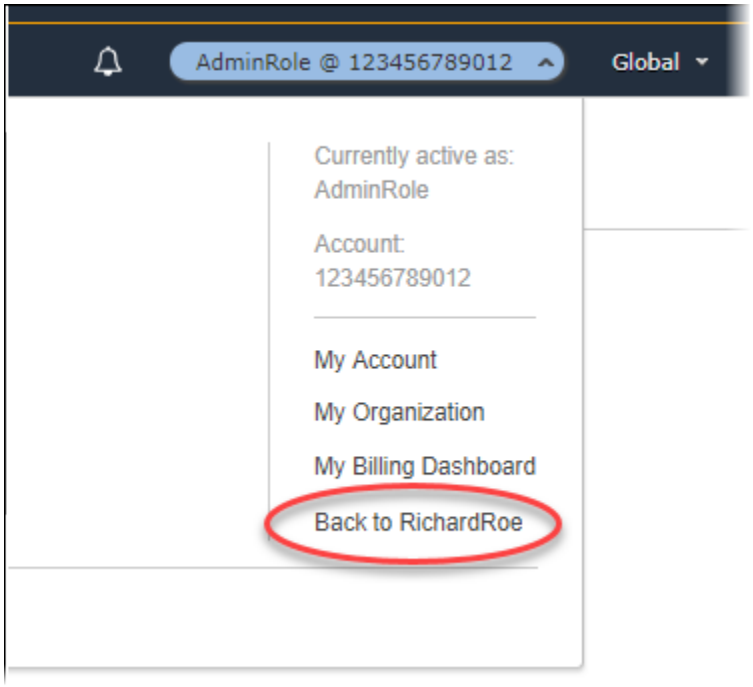
6. Wählen Sie Switch Role. Ihr Benutzername auf der Navigationsleiste wird durch den Anzeigenamen in der gewählten Farbe ersetzt und Sie können die durch die Rolle gewährten Berechtigungen nutzen.

> **Tipp**
>
> Die zuletzt verwendeten Rollen werden im Menü angezeigt. Wenn Sie erneut zu einer dieser Rollen wechseln möchten, können Sie einfach die gewünschte Rolle auswählen. Sie müssen die Konto- und Rolleninformationen nur manuell eingeben, wenn die Rolle nicht im Menü „Identity“ angezeigt wird.

So beenden Sie die Verwendung einer Rolle (Konsole)

1. Klicken Sie in der IAM-Konsole oben rechts in der Navigationsleiste auf Display Name (Anzeigename) der Rolle. Dieser hat normalerweise folgendes Format: ***Rollenname***@***Konto_ID_Nummer_oder_Alias***.
2. Wählen Sie Back to ***username*** (Zurück zu Benutzername). Die Rolle und ihre Berechtigungen werden deaktiviert und die Ihrem IAM-Benutzer und Ihren Gruppen zugeordneten Berechtigungen werden automatisch wiederhergestellt.

   Nehmen wir zum Beispiel an, dass Sie mit der Kontonummer `123456789012` und dem Benutzernamen `RichardRoe` angemeldet sind. Nach der Verwendung der `AdminRole`-Rolle können Sie die Verwendung der Rolle beenden und zu Ihren ursprünglichen Berechtigungen zurückkehren. Um die Verwendung einer Rolle zu beenden, wählen Sie AdminRole @ 123456789012 und dann Zurück zu. RichardRoe



Wechseln zu einer IAM-Rolle (AWS CLI)

Eine Rolle legt eine Gruppe von Berechtigungen fest, die Sie für den Zugriff auf AWS -Ressourcen, die Sie benötigen, verwenden können. In dieser Hinsicht ist sie mit einem IAM-Benutzer AWS Identity and Access Management vergleichbar. Wenn Sie sich als Benutzer anmelden, erhalten Sie einen bestimmten Satz von Berechtigungen. Sie melden sich jedoch nicht bei einer Rolle an. Wenn Sie sich aber als Benutzer angemeldet haben, können Sie zu einer Rolle wechseln. Dadurch werden Ihre ursprünglichen Benutzerberechtigungen vorübergehend zurückgestellt und Sie erhalten stattdessen die der Rolle zugewiesenen Berechtigungen. Die Rolle kann sich in Ihrem eigenen Konto oder jedem anderen AWS-Konto befinden. Weitere Informationen zu Rollen, ihren Vorteilen sowie zu ihrer Erstellung und Konfiguration finden Sie unter IAM-Rollenund Erstellen von IAM-Rollen. Weitere Informationen zu den unterschiedlichen Methoden, die Sie zum Übernehmen einer Rolle verwenden können, finden Sie unter Verwenden von IAM-Rollen.

> **Important**
>
> Die Berechtigungen Ihres IAM-Benutzers und alle Rollen, die Sie annehmen, können nicht kumuliert werden. Es ist nur jeweils ein Satz von Berechtigungen aktiv. Wenn Sie eine Rolle annehmen, geben Sie Ihre vorherigen Benutzer- oder Rollenberechtigungen temporär auf und arbeiten mit den Berechtigungen, die der Rolle zugeordnet sind. Wenn Sie die Rolle verlassen, werden Ihre Benutzerberechtigungen automatisch wiederhergestellt.

Sie können eine Rolle verwenden, um einen AWS CLI Befehl auszuführen, wenn Sie als IAM-Benutzer angemeldet sind. Sie können eine Rolle auch verwenden, um einen AWS CLI Befehl auszuführen, wenn Sie als extern authentifizierter Benutzer (SAML oder OIDC) angemeldet sind, der bereits eine Rolle verwendet. Darüber hinaus können Sie eine Rolle für die Ausführung eines AWS CLI -Befehls aus einer Amazon EC2-Instance verwenden, die einer Rolle über ihr Instance-Profil zugeordnet ist. Sie können keine Rolle anwenden, wenn Sie sich als Root-Benutzer des AWS-Kontos anmelden.

Verketten von Rollen – Sie können auch das Verketten von Rollen verwenden, was bedeutet, Berechtigungen einer Rolle für den Zugriff auf eine zweite Rolle zu verwenden.

Standardmäßig ist Ihre Rollensitzung eine Stunde gültig. Wenn Sie diese Rolle mithilfe der `assume-role*`-CLI-Operationen annehmen, können Sie einen Wert für den `duration-seconds`-Parameter angeben. Dieser Wert kann zwischen 900 Sekunden (15 Minuten) und der maximalen Sitzungsdauer für die Rolle liegen. Wenn Sie in der Konsole die Rollen wechseln, ist Ihre Sitzungsdauer auf maximal eine Stunde begrenzt. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.

Wenn Sie die Verkettung von Rollen verwenden, ist Ihre Sitzungsdauer auf die maximale Dauer von einer Stunde begrenzt. Wenn Sie den `duration-seconds`-Parameter verwenden, um einen Wert größer als eine Stunde anzugeben, schlägt die Operation fehl.

Beispielszenario: Wechseln zu einer Produktionsrolle

Stellen Sie sich vor, Sie sind ein IAM-Benutzer und arbeiten in der der Entwicklungsumgebung. In diesem Szenario müssen Sie gelegentlich mit der Produktionsumgebung an der Befehlszeile mit der AWS CLI arbeiten. Sie haben bereits einen Zugriffsschlüsselsatz, die Ihnen zur Verfügung steht. Dies kann das Zugriffsschlüsselpaar sein, das dem IAM-Standardbenutzer zugewiesen ist. Wenn Sie dagegen als Verbundbenutzer angemeldet sind, kann es sich um das Zugriffsschlüsselpaar für die Rolle handeln, die Ihnen ursprünglich zugewiesen wurde. Wenn Ihre aktuellen Berechtigungen Ihnen die Möglichkeit geben, eine bestimmte IAM-Rolle anzunehmen, können Sie diese Rolle in einem „Profil“ in den Konfigurationsdateien identifizieren. AWS CLI Dieser Befehl wird dann mit den Berechtigungen der angegebenen IAM-Rolle, nicht mit der ursprünglichen Identität ausgeführt. Beachten Sie, dass Sie die neue Rolle verwenden, wenn Sie dieses Profil in einem AWS CLI Befehl angeben. In dieser Situation können Sie nicht gleichzeitig die ursprünglichen Berechtigungen im Entwicklungskonto nutzen. Der Grund besteht darin, dass zu einem bestimmten Zeitpunkt nur jeweils ein Satz Berechtigungen wirksam sein kann.

> **Note**
>
> Aus Sicherheitsgründen können Administratoren anhand von AWS CloudTrail Protokollen herausfinden, wer eine Aktion in ausgeführt hat AWS. Ihr Administrator erfordert möglicherweise, dass Sie einen bestimmten Wert für den Sitzungsnamen angeben, wenn Sie die Rolle übernehmen. Weitere Informationen finden Sie unter `sts:SourceIdentity` und `sts:RoleSessionName`.

So wechseln Sie zu einer Produktionsrolle (AWS CLI)

1. Wenn Sie das noch nie verwendet haben AWS CLI, müssen Sie zuerst Ihr Standard-CLI-Profil konfigurieren. Öffnen Sie eine Befehlszeile und richten Sie Ihre AWS CLI Installation so ein, dass sie den Zugriffsschlüssel Ihres IAM-Benutzers oder Ihrer Verbundrolle verwendet. Weitere Informationen finden Sie unter Konfigurieren der AWS Command Line Interface im AWS Command Line Interface -Leitfaden.

   Konfigurieren Sie die mit dem Befehl wie folgt:

   ```
   aws configure
   ```

   Wenn Sie dazu aufgefordert werden, geben Sie die folgenden Informationen an:

   ```
   AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   Default region name [None]: us-east-2
   Default output format [None]: json
   ```

2. Erstellen Sie ein neues Profil für die Rolle in der Datei `.aws/config` in Unix oder Linux oder in der Datei `C:\Users\USERNAME\.aws\config` in Windows. Im folgenden Beispiel wird ein Profil mit dem Namen `prodaccess` erstellt, das zur Rolle *`ProductionAccessRole`* im `123456789012`-Konto wechselt. Der ARN der Rolle wird Ihnen vom Kontoadministrator, der die Rolle erstellt hat, mitgeteilt. Wenn dieses Profil aufgerufen wird, AWS CLI verwendet es die Anmeldeinformationen von, um Anmeldeinformationen für die `source_profile` Rolle anzufordern. Daher muss die Identität, die als `source_profile` referenziert ist, über `sts:AssumeRole`-Berechtigungen für die Rolle verfügen, die in `role_arn` angegeben ist.

   ```
   [profile prodaccess]
       role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
   ```

```
    source_profile = default
```

3. Nachdem Sie das neue Profil erstellt haben, wird jeder AWS CLI Befehl, der den Parameter angibt, unter den Berechtigungen `--profile prodaccess` ausgeführt, die der IAM-Rolle zugewiesen sind, und nicht unter den Berechtigungen `ProductionAccessRole` des Standardbenutzers.

   ```
   aws iam list-users --profile prodaccess
   ```

   Dieser Befehl funktioniert, wenn die Berechtigungen, die `ProductionAccessRole` zugeordnet sind, das Auflisten der Benutzer im aktuellen AWS -Konto ermöglichen.

4. Um zu den Berechtigungen zurückzukehren, die von Ihren ursprünglichen Anmeldeinformationen erteilt wurden, führen Sie Befehle ohne den `--profile`-Parameter. Das AWS CLI verwendet wieder die Anmeldeinformationen in Ihrem Standardprofil, in dem Sie es konfiguriert haben. Step 1

Weitere Informationen hierzu finden Sie unter Assuming a Role im AWS Command Line Interface Leitfaden.

Beispielszenario: Einer Instance-Profilrolle erlauben, zu einer Rolle in einem anderen Konto zu wechseln

Stellen Sie sich vor AWS-Konten, Sie verwenden zwei und möchten einer Anwendung, die auf einer Amazon EC2 EC2-Instance ausgeführt wird, erlauben, AWS CLIBefehle in beiden Konten auszuführen. Gehen Sie davon aus, dass die EC2-Instance im Konto `111111111111` vorhanden ist. Diese Instance enthält die `abcd`-Instance-Profilrolle, die der Anwendung erlaubt, schreibgeschützte Amazon S3-Aufgaben für den `my-bucket-1`-Bucket innerhalb desselben `111111111111` Kontos auszuführen. Die Anwendung muss aber auch berechtigt sein, die kontoübergreifende `efgh`-Rolle zu übernehmen, um Aufgaben im Konto `222222222222` auszuführen. Dazu muss der EC2-Instance-Profilrolle `abcd` die folgende Berechtigungsrichtlinie zugeordnet sein.

Konto 111111111111 ***abcd*** Rollen-Berechtigungsrichtlinie

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Gehen Sie davon aus, dass die kontoübergreifende `efgh`-Rolle die Ausführung schreibgeschützter Amazon S3-Aufgaben für den `my-bucket-2`-Bucket im selben `222222222222`-Konto zulässt. Dazu muss der kontoübergreifenden `efgh`-Rolle die folgende Berechtigungsrichtlinie zugeordnet sein:

Konto 222222222222 ***efgh*** Rollen-Berechtigungsrichtlinie

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

Die `efgh`-Rolle muss der `abcd`-Instance-Profilrolle die Übernahme erlauben. Dazu benötigt die `efgh`-Rolle die folgende Vertrauensrichtlinie:

Konto 222222222222 ***`efgh`*** Rollen-Berechtigungsrichtlinie

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Um dann AWS CLI Befehle im Konto auszuführen222222222222, müssen Sie die CLI-Konfigurationsdatei aktualisieren. Identifizieren Sie die `efgh`-Rolle als "Profil" und die `abcd`-EC2-Instance-Profilrolle als "Quelle der Anmeldeinformationen“ in der AWS CLI -Konfigurationsdatei.

Dann werden Ihre CLI-Befehle mit den Berechtigungen der `efgh`-Rolle und nicht mit denen der ursprünglichen `abcd`-Rolle ausgeführt.

> **Note**
>
> Aus Sicherheitsgründen können Sie AWS CloudTrail damit die Verwendung von Rollen im Konto überprüfen. Um in CloudTrail Protokollen zwischen Rollensitzungen zu unterscheiden, wenn eine Rolle von verschiedenen Prinzipalen verwendet wird, können Sie den Namen der Rollensitzung verwenden. Wenn der eine Rolle im Namen eines Benutzers AWS CLI übernimmt, wie in diesem Thema beschrieben, wird automatisch ein Rollensitzungsname als `AWS-CLI-session-`*`nnnnnnnn`* erstellt. Hier ist *`nnnnnnnn`* eine Ganzzahl, die die Zeit in Unix-Epoche (die Anzahl der Sekunden seit Mitternacht UTC am 1. Januar 1970) angibt. Weitere Informationen finden Sie unter CloudTrail Event Reference im AWS CloudTrail Benutzerhandbuch.

So lassen Sie zu, dass eine EC2-Instance-Profilrolle zu einer kontoübergreifenden Rolle wechselt (AWS CLI)

1. Sie müssen kein Standard-CLI-Profil konfigurieren. Stattdessen können Sie Anmeldeinformationen aus den Metadaten des EC2-Instance-Profils laden. Erstellen Sie ein neues Profil für die Rolle in der `.aws/config`-Datei. Das folgende Beispiel erstellt ein `instancecrossaccount`-Profil, das zur *`efgh`*-Rolle im 222222222222-Konto wechselt. Wenn dieses Profil aufgerufen wird, verwendet die AWS CLI die Anmeldeinformationen aus den Metadaten des EC2-Instance-Profils, um Anmeldeinformationen für die Rolle anzufordern. Deshalb muss das EC2-Instance-Profil über die `sts:AssumeRole`-Berechtigungen für die Rolle verfügen, die in `role_arn` angegeben ist.

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. Nachdem Sie das neue Profil erstellt haben, wird jeder AWS CLI Befehl, der den Parameter angibt, unter den Berechtigungen `--profile instancecrossaccount` ausgeführt, die der `efgh` Rolle im Konto zugewiesen sind222222222222.

   ```
   aws s3 ls my-bucket-2 --profile instancecrossaccount
   ```

Dieser Befehl funktioniert, wenn die Berechtigungen, die zur `efgh`-Rolle zugewiesen sind, die Auflistung der Benutzer im aktuellen AWS-Konto zulässt.

3. Zum Zurückkehren zu den ursprünglichen EC2-Instance-Profilberechtigungen im Konto `111111111111` führen Sie die CLI-Befehle ohne den Parameter `--profile`.

Weitere Informationen hierzu finden Sie unter Assuming a Role im AWS Command Line Interface Leitfaden.

Zu einer IAM-Rolle wechseln (Tools für Windows PowerShell)

Eine Rolle legt eine Gruppe von Berechtigungen fest, die Sie für den Zugriff auf AWS -Ressourcen, die Sie benötigen, verwenden können. In dieser Hinsicht ist sie mit einem IAM-Benutzer AWS Identity and Access Management vergleichbar. Wenn Sie sich als Benutzer anmelden, erhalten Sie einen bestimmten Satz von Berechtigungen. Sie melden sich nicht bei einer Rolle an, aber sobald Sie angemeldet sind, können Sie zu einer Rolle wechseln. Dadurch werden Ihre ursprünglichen Benutzerberechtigungen vorübergehend zurückgestellt und Sie erhalten stattdessen die der Rolle zugewiesenen Berechtigungen. Die Rolle kann sich in Ihrem eigenen Konto oder jedem anderen AWS-Konto befinden. Weitere Informationen zu Rollen, ihren Vorteilen sowie zu ihrer Erstellung und Konfiguration finden Sie unter IAM-Rollenund Erstellen von IAM-Rollen.

> **Important**
>
> Die Berechtigungen Ihres IAM-Benutzers und alle Rollen, zu denen Sie wechseln, können nicht kumuliert werden. Es ist nur jeweils ein Satz von Berechtigungen aktiv. Wenn Sie zu einer Rolle wechseln, geben Sie Ihre Benutzerberechtigungen temporär auf und arbeiten mit den Berechtigungen, die der Rolle zugeordnet sind. Wenn Sie die Rolle verlassen, werden Ihre Benutzerberechtigungen automatisch wiederhergestellt.

In diesem Abschnitt wird beschrieben, wie Sie Rollen wechseln, wenn Sie mit den AWS Tools for Windows PowerShell in der Befehlszeile arbeiten.

Stellen Sie sich vor, Sie haben ein Konto in der Entwicklungsumgebung und müssen gelegentlich über die Befehlszeile mit den Tools für Windows PowerShell mit der Produktionsumgebung arbeiten. Es steht Ihnen bereits ein Satz an Zugriffsschlüsseln zur Verfügung. Dies kann ein Zugriffsschlüsselpaar sein, das dem IAM-Standardbenutzer zugewiesen ist. Wenn Sie dagegen als Verbundbenutzer angemeldet sind, kann es sich um das Zugriffsschlüsselpaar für die Rolle handeln,

die Ihnen ursprünglich zugewiesen wurde. Sie können diese Anmeldeinformationen verwenden, um das `Use-STSRole`-Cmdlet auszuführen, das den ARN einer neuen Rolle als Parameter übergibt. Der Befehl gibt temporäre Sicherheitsanmeldeinformationen für die angeforderte Rolle zurück. Sie können diese Anmeldeinformationen dann in nachfolgenden PowerShell Befehlen mit den Berechtigungen der Rolle für den Zugriff auf Ressourcen in der Produktion verwenden. Solange Sie die Rolle verwenden, können Sie Ihre Benutzerberechtigungen im Entwicklungskonto nicht nutzen, da immer nur jeweils ein Satz an Berechtigungen wirksam ist.

> **Note**
>
> Aus Sicherheitsgründen können Administratoren anhand von AWS CloudTrail Protokollen herausfinden, wer eine Aktion in ausgeführt hat AWS. Ihr Administrator erfordert möglicherweise, dass Sie einen bestimmten Wert für den Sitzungsnamen angeben, wenn Sie die Rolle übernehmen. Weitere Informationen finden Sie unter `sts:SourceIdentity` und `sts:RoleSessionName`.

Beachten Sie, dass alle Zugriffsschlüssel und Token nur Beispiele sind und nicht wie hier dargestellt verwendet werden können. Ersetzen Sie sie mit den entsprechenden Werten aus Ihrer Live-Umgebung.

Um zu einer Rolle zu wechseln (Tools für Windows PowerShell)

1. Öffnen Sie eine PowerShell Befehlszeile und konfigurieren Sie das Standardprofil so, dass es den Zugriffsschlüssel Ihres aktuellen IAM-Benutzers oder Ihrer Verbundrolle verwendet. Wenn Sie die Tools für Windows bereits verwendet haben PowerShell, ist dies wahrscheinlich bereits geschehen. Beachten Sie, dass Sie Rollen nur dann wechseln können, wenn Sie als IAM-Benutzer und nicht als der Root-Benutzer des AWS-Kontos angemeldet sind.

   ```
   PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
   SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
   PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
   ```

   Weitere Informationen finden Sie im AWS Tools for Windows PowerShell Benutzerhandbuch unter Verwenden von AWS Anmeldeinformationen.
2. Um die Anmeldeinformationen für die neue Rolle abzurufen, führen Sie den folgenden Befehl aus, um zur *RoleName*-Rolle im Konto 123456789012 zu wechseln. Der ARN der Rolle wird Ihnen vom Kontoadministrator, der die Rolle erstellt hat, mitgeteilt. Der Befehl erfordert, dass

Sie außerdem einen Sitzungsnamen angeben. Sie können hierfür einen beliebigen Text wählen. Der folgende Befehl fordert die Anmeldeinformationen an und erfasst dann das `Credentials`-Eigenschaftsobjekt vom zurückgegebenen Ergebnisobjekt und speichert es in der `$Creds`-Variablen.

```
PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
RoleSessionName "MyRoleSessionName").Credentials
```

`$Creds` ist ein Objekt, das jetzt die Elemente `AccessKeyId`, `SecretAccessKey` und `SessionToken` enthält, die Sie in den folgenden Schritten benötigen. Die folgenden Beispielbefehle veranschaulichen typische Werte:

```
PS C:\> $Creds.AccessKeyId
AKIAIOSFODNN7EXAMPLE

PS C:\> $Creds.SecretAccessKey
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

PS C:\> $Creds.SessionToken
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Um diese Anmeldeinformationen für nachfolgende Befehle zu verwenden, fügen Sie sie dem `-Credential`-Parameter bei. Der folgende Befehl verwendet beispielsweise die Anmeldeinformationen der Rolle und funktioniert nur dann, wenn der Rolle die `iam:ListRoles`-Berechtigung gewährt wird und sie daher das `Get-IAMRoles`-Cmdlet ausführen kann:

```
PS C:\> get-iamroles -Credential $Creds
```

4. Um zu Ihren ursprünglichen Anmeldeinformationen zurückzukehren, beenden Sie einfach die Verwendung des `-Credentials $Creds` Parameters und erlauben Sie PowerShell , zu den Anmeldeinformationen zurückzukehren, die im Standardprofil gespeichert sind.

Zu einer IAM-Rolle (AWS API) wechseln

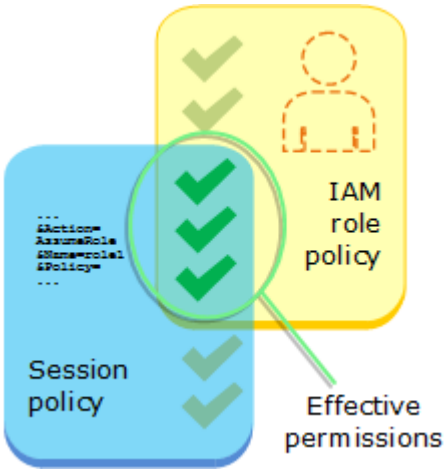
Eine Rolle legt eine Gruppe von Berechtigungen fest, die Sie für den Zugriff auf AWS -Ressourcen verwenden können. In dieser Hinsicht ist sie mit einem IAM-Benutzer vergleichbar. Ein Principal (Person oder Anwendung) übernimmt eine Rolle, um temporäre Berechtigungen zur Ausführung erforderlicher Aufgaben und zur Interaktion mit AWS Ressourcen zu erhalten. Die Rolle kann sich in Ihrem eigenen Konto oder jedem anderen AWS-Konto befinden. Weitere Informationen zu Rollen, ihren Vorteilen sowie zu ihrer Erstellung und Konfiguration finden Sie unter IAM-Rollenund Erstellen von IAM-Rollen. Weitere Informationen zu den unterschiedlichen Methoden, die Sie zum Übernehmen einer Rolle verwenden können, finden Sie unter Verwenden von IAM-Rollen.

> **Important**
>
> Die Berechtigungen Ihres IAM-Benutzers und alle Rollen, die Sie annehmen, können nicht kumuliert werden. Es ist nur jeweils ein Satz von Berechtigungen aktiv. Wenn Sie eine Rolle annehmen, geben Sie Ihre vorherigen Benutzer- oder Rollenberechtigungen temporär auf und arbeiten mit den Berechtigungen, die der Rolle zugeordnet sind. Wenn Sie die Rolle verlassen, werden Ihre ursprünglichen Benutzerberechtigungen automatisch wiederhergestellt.

Um eine Rolle anzunehmen, ruft eine Anwendung den AWS STS `AssumeRole`API-Vorgang auf und übergibt den ARN der zu verwendenden Rolle. Die Operation erstellt eine neue Sitzung mit temporären Anmeldeinformationen. Diese Sitzung hat die gleichen Berechtigungen wie die identitätsbasierten Richtlinien für diese Rolle.

Wenn Sie `AssumeRole aufrufen`, können Sie optional eingebundene oder verwaltete Sitzungsrichtlinien übergeben. Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie programmgesteuert eine temporäre Anmeldeinformationen-Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Sie können ein einzelnes JSON-Inline-Sitzungsrichtliniendokument mit dem Parameter `Policy` übergeben. Sie können mit dem Parameter `PolicyArns` bis zu 10 verwaltete Sitzungsrichtlinien angeben. Die resultierenden Sitzungsberechtigungen bilden die Schnittmenge der auf der Identität der Entity basierenden

Richtlinien und der Sitzungsrichtlinien. Sitzungsrichtlinien sind praktisch, wenn Sie die temporären Anmeldeinformationen einer Rolle einer anderen Person übergeben müssen. Sie können die temporären Anmeldeinformationen der Rolle in nachfolgenden AWS -API-Aufrufen verwenden, um auf Ressourcen in dem Konto zuzugreifen, zu dem die Rolle gehört. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtline zulässig sind. Weitere Informationen darüber, wie die effektiven Berechtigungen einer Rolle AWS bestimmt werden, finden Sie unterAuswertungslogik für Richtlinien.



Sie können `AssumeRole` aufrufen, wenn Sie als IAM-Benutzer oder als extern authentifizierter Benutzer (SAML oder OIDC), der eine Rolle bereits verwendet, angemeldet sind. Sie können auch das Verketten von Rollen verwenden, was bedeutet, eine Rolle zu verwenden, um eine andere Rolle anzunehmen. Sie können keine Rolle anwenden, wenn Sie sich als Root-Benutzer des AWS-Kontos anmelden.

Standardmäßig ist Ihre Rollensitzung eine Stunde gültig. Wenn Sie diese Rolle mithilfe der AWS STS `AssumeRole*`API-Operationen übernehmen, können Sie einen Wert für den `DurationSeconds` Parameter angeben. Dieser Wert kann zwischen 900 Sekunden (15 Minuten) und der maximalen Sitzungsdauer für die Rolle liegen. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.

Wenn Sie die Verkettung von Rollen verwenden, ist Ihre Sitzung auf die maximale Dauer von einer Stunde begrenzt. Wenn Sie den `DurationSeconds`-Parameter verwenden, um einen Wert größer als eine Stunde anzugeben, schlägt die Operation fehl.

> **Note**
>
> Aus Sicherheitsgründen können Administratoren anhand von AWS CloudTrail Protokollen herausfinden, wer eine Aktion in ausgeführt hat AWS. Ihr Administrator erfordert

möglicherweise, dass Sie einen bestimmten Wert für den Sitzungsnamen angeben, wenn Sie die Rolle übernehmen. Weitere Informationen finden Sie unter `sts:SourceIdentity` und `sts:RoleSessionName`.

Die folgenden Codebeispiele veranschaulichen, wie Sie einen Benutzer erstellen und eine Rolle annehmen lassen.

> **Warning**
>
> Um Sicherheitsrisiken zu vermeiden, sollten Sie IAM-Benutzer nicht zur Authentifizierung verwenden, wenn Sie eigens entwickelte Software entwickeln oder mit echten Daten arbeiten. Verwenden Sie stattdessen den Verbund mit einem Identitätsanbieter wie AWS IAM Identity Center.

- Erstellen Sie einen Benutzer ohne Berechtigungen.
- Erstellen einer Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets für das Konto erteilt.
- Hinzufügen einer Richtlinie, damit der Benutzer die Rolle übernehmen kann.
- Übernehmen Sie die Rolle und listen Sie S3-Buckets mit temporären Anmeldeinformationen auf, und bereinigen Sie dann die Ressourcen.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
```

```
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
```

```
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
```

```
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };
```

```
        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
```

```
   /// access key.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
   {
       var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
       {
           AccessKeyId = accessKeyId,
           UserName = userName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Delete an IAM group.
   /// </summary>
   /// <param name="groupName">The name of the IAM group to delete.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteGroupAsync(string groupName)
   {
       var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
       return response.HttpStatusCode == HttpStatusCode.OK;
   }


   /// <summary>
   /// Delete an IAM policy associated with an IAM group.
   /// </summary>
   /// <param name="groupName">The name of the IAM group associated with the
   /// policy.</param>
   /// <param name="policyName">The name of the policy to delete.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
   {
       var request = new DeleteGroupPolicyRequest()
       {
           GroupName = groupName,
           PolicyName = policyName,
       };
```

```
        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
```

```
        var response = await _IAMService.DeleteRolePolicyAsync(new
DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
```

```
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
 policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }
```

```
    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });
```

```
        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
```

```
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
```

```
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
 _IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
```

```
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
```

```
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
```

```
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
```

```
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                $" \"AWS\": \"{userArn}\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
            "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
```

```
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);
```

```
        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);
```

```
        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
```

```
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
```

```
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();
```

```
        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
(IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
```

```
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }
```

```
        PressEnter();
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
```

```
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
```

```
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi
```

```
  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
```

```
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret

  local buckets
  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. This should not have
happened."
  else
    errecho "Because the role with permissions has not been assumed, listing
buckets failed."
  fi

  echo
  echo_repeat "*" 88
  echo "Now assume the role $iam_role_name and list the buckets."
  echo_repeat "*" 88
  echo
```

```
  local credentials

  credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Assumed role $iam_role_name"
  else
    errecho "Failed to assume role."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  IFS=$'\t ' read -r -a credentials <<<"$credentials"

  export AWS_ACCESS_KEY_ID=${credentials[0]}
  export AWS_SECRET_ACCESS_KEY=${credentials[1]}
  # bashsupport disable=BP2001
  export AWS_SESSION_TOKEN=${credentials[2]}

  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""
```

```
  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Die in diesem Szenario verwendeten IAM-Funktionen.

```
###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
```

```
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }
```

```
# Retrieve the calling parameters.
while getopts "u:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name already exists in the account."
  return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
  --output text \
  --query 'User.Arn')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-user operation failed.$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
```

```
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
```

```
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
```

```
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
```

```
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
```

```
}

################################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
```

```
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
```

```
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

################################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
```

```
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # bashsupport disable=BP5008
 function usage() {
   echo "function iam_delete_user"
   echo "Deletes an WS Identity and Access Management (IAM) user. You must
supply a username:"
   echo "  -u user_name    The name of the user."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "u:h" option; do
   case "${option}" in
     u) user_name="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$user_name" ]]; then
   errecho "ERROR: You must provide a username with the -u parameter."
   usage
   return 1
 fi

 iecho "Parameters:\n"
 iecho "    User name:   $user_name"
 iecho ""

 # If the user does not exist, we don't want to try to delete it.
 if (! iam_user_exists "$user_name"); then
   errecho "ERROR: A user with that name does not exist in the account."
   return 1
 fi

 response=$(aws iam delete-user \
```

```
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- API-Details finden Sie in den folgenden Themen der AWS CLI -Befehlsreferenz.
  - AttachRoleRichtlinie
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
```

```
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
```

```
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

    request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
```

```
                    outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;
```

```
                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
```

```
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
```

```
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
```

```
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
```

```
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for C++ -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik

- PutUserPolitik

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
// 1. Create a user who has no permissions.
// 2. Create a role that grants permission to list Amazon Simple Storage Service
//    (Amazon S3) buckets for the account.
// 3. Add a policy to let the user assume the role.
// 4. Try and fail to list buckets without permissions.
// 5. Assume the role and list S3 buckets using temporary credentials.
// 6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper actions.PolicyWrapper
 roleWrapper actions.RoleWrapper
 userWrapper actions.UserWrapper
 questioner demotools.IQuestioner
 helper IScenarioHelper
 isTestRun bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
```

```
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
  helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:   sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
```

```
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser() *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}
```

```
// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(user *types.User)
 *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err :=
 scenario.roleWrapper.CreateRole(scenario.helper.GetName(), *user.Arn)
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"}, "arn:aws:s3:::*")
 if err != nil {panic(err)}
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(*listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {panic(err)}
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(*user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {panic(err)}
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
```

```
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(accessKey
 *types.AccessKey) *aws.Config {
  log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
  scenario.questioner.Ask("Press Enter when you're ready.")
  noPermsConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
 ))
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

  s3Client := s3.NewFromConfig(noPermsConfig)
  _, err = s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
  } else {
   log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
    "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
  return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
```

```
// 1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//   the user's access key credentials.
// 2. Gets temporary credentials by assuming the role that grants permission to
 list the
//    buckets.
// 3. Creates an Amazon S3 client from the temporary credentials.
// 4. Lists buckets for the account. Because the temporary credentials are
 generated by
//    assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(noPermsConfig
 *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(context.TODO(),
 &sts.AssumeRoleInput{
  RoleArn:           role.Arn,
  RoleSessionName:   aws.String("AssumeRoleExampleSession"),
  DurationSeconds:   aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {panic(err)}

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
```

```
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(user *types.User, role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
   policies, err := scenario.roleWrapper.ListAttachedRolePolicies(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*policy.PolicyArn)
    if err != nil {panic(err)}
    log.Printf("Detached policy %v from role %v and deleted the policy.\n",
     *policy.PolicyName, *role.RoleName)
   }
   err = scenario.roleWrapper.DeleteRole(*role.RoleName)
   if err != nil {panic(err)}
   log.Printf("Deleted role %v.\n", *role.RoleName)

   userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
   if err != nil {panic(err)}
   for _, userPol := range userPols {
    err = scenario.userWrapper.DeleteUserPolicy(*user.UserName, userPol)
    if err != nil {panic(err)}
    log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
   }
   keys, err := scenario.userWrapper.ListAccessKeys(*user.UserName)
   if err != nil {panic(err)}
   for _, key := range keys {
    err = scenario.userWrapper.DeleteAccessKey(*user.UserName, *key.AccessKeyId)
    if err != nil {panic(err)}
```

```
    log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
   }
   err = scenario.userWrapper.DeleteUser(*user.UserName)
   if err != nil {panic(err)}
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}
```

Definieren Sie eine Struktur, die Kontoaktionen umschließt.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

```
// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Definieren Sie eine Struktur, die Richtlinienaktionen umschließt.

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect string
 Action []string
 Principal map[string]string `json:",omitempty"`
 Resource *string `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
```

```
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
```

```
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Definieren Sie eine Struktur, die Rollenaktionen umschließt.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
```

```
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
```

```
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

```
// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
```

```
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Definieren Sie eine Struktur, die Benutzeraktionen umschließt.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
```

```
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
```

```
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
```

```
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Go -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Benutzer-Aktionen umschließen.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the AWS IAM example scenario.");
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();
```

```
            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }
```

```
            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();
```

```
            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);
```

```
            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
```

```
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer und eine Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen. Nachdem Sie die Rolle übernommen haben, verwenden Sie temporäre Anmeldeinformationen, um Buckets für das Konto aufzulisten.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
```

```
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    } else {
      throw caught;
    }
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );
```

```
  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
```

```
            {
              Effect: "Allow",
              Principal: {
                // Allow the previously created user to assume this role.
                AWS: User.Arn,
              },
              Action: "sts:AssumeRole",
            },
          ],
        }),
        RoleName: roleName,
      }),
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
```

```
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 60 }, () =>
  listBuckets(s3Client),
```

```
  );

  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
```

```
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for JavaScript -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Benutzer-Aktionen umschließen.

```
suspend fun main(args: Array<String>) {
```

```
   val usage = """
   Usage:
       <username> <policyName> <roleName> <roleSessionName> <fileLocation>
<bucketName>

   Where:
       username - The name of the IAM user to create.
       policyName - The name of the policy to create.
       roleName - The name of the role to create.
       roleSessionName - The name of the session required for the assumeRole
operation.
       fileLocation - The file location to the JSON required to create the role
(see Readme).
       bucketName - The name of the Amazon S3 bucket from which objects are
read.
   """

   if (args.size != 6) {
       println(usage)
       exitProcess(1)
   }

   val userName = args[0]
   val policyName = args[1]
   val roleName = args[2]
   val roleSessionName = args[3]
   val fileLocation = args[4]
   val bucketName = args[5]

   createUser(userName)
   println("$userName was successfully created.")

   val polArn = createPolicy(policyName)
   println("The policy $polArn was successfully created.")

   val roleArn = createRole(roleName, fileLocation)
   println("$roleArn was successfully created.")
   attachRolePolicy(roleName, polArn)

   println("*** Wait for 1 MIN so the resource is available.")
   delay(60000)
   assumeGivenRole(roleArn, roleSessionName, bucketName)

   println("*** Getting ready to delete the AWS resources.")
```

```
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue: String =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

```
suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
```

```
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String
) {
    val stsClient =
        StsClient {
            region = "us-east-1"
        }

    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken
```

```
    val staticCredentials =
        StaticCredentialsProvider {
            accessKeyId = key
            secretAccessKey = secKey
            sessionToken = secToken
        }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 =
        S3Client {
            credentialsProvider = staticCredentials
            region = "us-east-1"
        }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String
) {
    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)
```

```
    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.
  - AttachRolePolitik

- CreateAccessSchlüssel
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessSchlüssel
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");
```

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
```

```
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for PHP -API-Referenz.

- AttachRolePolitik
- CreateAccessSchlüssel
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessSchlüssel
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer und eine Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen. Nachdem Sie die Rolle übernommen haben, verwenden Sie temporäre Anmeldeinformationen, um Buckets für das Konto aufzulisten.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError
```

```
def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
```

```
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
"
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)
```

```
    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
```

```
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
```

```
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
```

```
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer und eine Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen. Nachdem Sie die Rolle übernommen haben, verwenden Sie temporäre Anmeldeinformationen, um Buckets für das Konto aufzulisten.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts("Give AWS time to propagate resources...")
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Tried and failed to create demo user.")
```

```
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
                    Effect: "Allow",
                    Principal: {'AWS': user.arn},
                    Action: "sts:AssumeRole"
                  }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Action: "s3:ListAllMyBuckets",
                   Resource: "arn:aws:s3:::*"
                 }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
```

```
 # @param role [Aws::IAM::Role] The role that can be assumed.
 # @return [Aws::IAM::UserPolicy] The newly created policy.
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: "2012-10-17",
     Statement: [{
                  Effect: "Allow",
                  Action: "sts:AssumeRole",
                  Resource: role.arn
                }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
```

```
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == "AccessDenied"
      puts("Attempt to list buckets with no permissions: AccessDenied.")
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
  # @param key_id [String] The ID of the access key used by the STS client.
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: "create-use-assume-role-scenario"
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

```
  # Deletes a role. If the role has policies attached, they are detached and
  # deleted before the role is deleted.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
 role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
 policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
```

```
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Ruby -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessSchlüssel
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
```

```
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
```

```
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
```

```
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
```

```
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
```

```
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zu AWS - SDK für Rust.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser

- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden

Anwendungen, die auf einer Amazon EC2 EC2-Instance ausgeführt werden, müssen AWS Anmeldeinformationen in den AWS API-Anfragen enthalten. Sie könnten Ihre Entwickler AWS Anmeldeinformationen direkt in der Amazon EC2 EC2-Instance speichern lassen und Anwendungen in dieser Instance erlauben, diese Anmeldeinformationen zu verwenden. Allerdings müssten Entwickler dann die Anmeldeinformationen verwalten und sicherstellen, dass sie die Anmeldeinformationen sicher an jede Instance weitergeben und jede Amazon-EC2-Instance aktualisieren, wenn es Zeit für die Aktualisierung der Anmeldeinformationen ist. Das ist eine Menge zusätzlicher Arbeit.

Stattdessen können und sollten Sie eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer Amazon-EC2-Instance ausgeführt werden. Wenn Sie eine Rolle verwenden, müssen Sie keine langfristigen Anmeldeinformationen (wie Benutzername und Passwort oder Zugriffsschlüssel) an eine Amazon-EC2-Instance verteilen. Stattdessen stellt die Rolle temporäre Berechtigungen bereit, die Anwendungen verwenden können, wenn sie andere AWS Ressourcen aufrufen. Wenn Sie eine Amazon EC2-Instance starten, geben Sie eine mit der Instance zu verknüpfende IAM-Rolle an. Anwendungen, die auf der Instance ausgeführt werden, können dann die von der Rolle bereitgestellten temporären Anmeldeinformationen für API-Anforderungen verwenden.

Die Verwendung von Rollen zur Erteilung von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden, erfordert etwas mehr Konfigurationsaufwand. Eine Anwendung, die auf einer Amazon EC2 EC2-Instance ausgeführt wird, wird AWS vom virtualisierten Betriebssystem abstrahiert. Aufgrund dieser zusätzlichen Trennung benötigen Sie einen zusätzlichen Schritt, um einer Amazon EC2 EC2-Instance eine AWS Rolle und die zugehörigen Berechtigungen zuzuweisen und sie für ihre Anwendungen verfügbar zu machen. Dieser zusätzliche Schritt ist die Erstellung eines Instance-Profils, das einer Instance angefügt wird. Das Instance-Profil enthält die Rolle und kann einer auf einer Instance ausgeführten Anwendung die temporären Anmeldeinformationen der Rolle zur Verfügung stellen. Diese temporären Anmeldeinformationen können dann für die API-Aufrufe der Anwendung verwendet werden, um auf Ressourcen zuzugreifen und den Zugriff auf die in der Rolle angegebenen Ressourcen zu beschränken.

> **Note**
>
> Einer Amazon-EC2-Instance kann jeweils nur eine Rolle zugewiesen werden und alle Anwendungen auf der Instance haben dieselbe Rolle und dieselben Berechtigungen. Wenn Sie Amazon ECS zur Verwaltung Ihrer Amazon-EC2-Instances nutzen, können Sie Amazon-ECS-Aufgaben Rollen zuweisen, die sich von der Rolle der Amazon-EC2-Instance unterscheiden lassen, 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.
> Weitere Informationen finden Sie unter Verwenden von IAM-Rollen mit Amazon-ECS-Aufgaben im Handbuch zu bewährten Methoden zu Amazon Elastic Container Service.

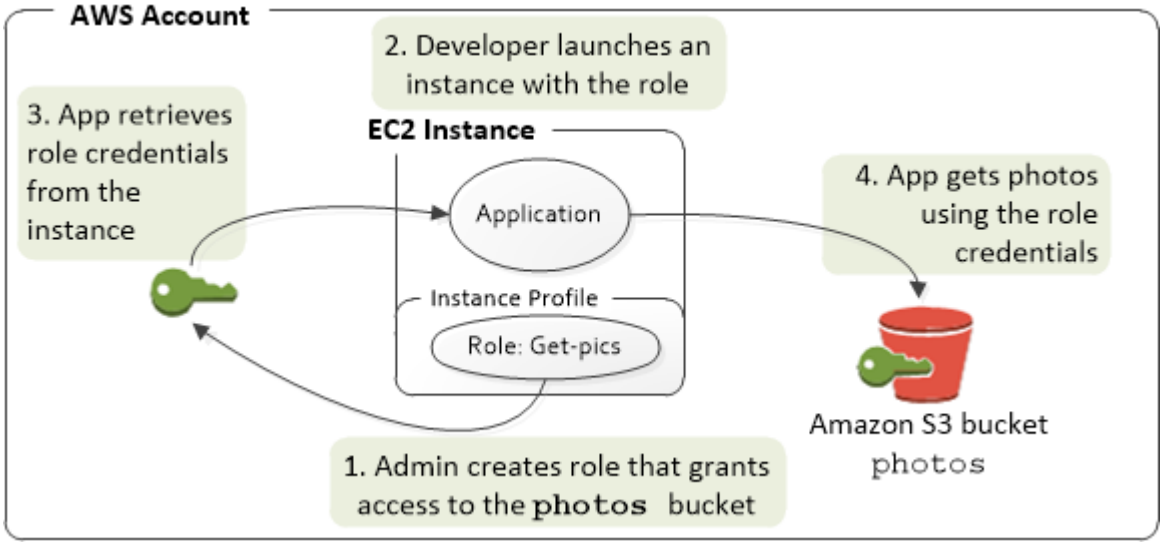
Diese Verwendungsweise von Rollen hat mehrere Vorteile. Da Rollenanmeldeinformationen temporär sind und automatisch aktualisiert werden, müssen Sie die Anmeldeinformationen nicht verwalten und sich keine Sorgen über langfristige Sicherheitsrisiken machen. Wenn Sie darüber hinaus eine einzelne Rolle für mehrere Instances verwenden, werden alle Änderungen in dieser Rolle automatisch an alle Instances propagiert.

> **Note**
>
> Obwohl eine Rolle normalerweise einer Amazon-EC2-Instance zugewiesen wird, wenn Sie sie starten, kann eine Rolle auch einer Amazon-EC2-Instance angehängt werden, diese gerade ausgeführt wird. Weitere Informationen zum Anfügen einer Rolle an eine ausgeführte Instance erhalten Sie unter IAM-Rollen für Amazon EC2.

Themen

Wie funktionieren Rollen für Amazon-EC2-Instances?

In der folgenden Abbildung führt ein Entwickler eine Anwendung auf einer Amazon-EC2-Instance aus, die Zugriff auf den S3-Bucket namens `photos` benötigt. Ein Administrator erstellt die Servicerolle `Get-pics` und weist die Rolle der Amazon-EC2-Instance zu. Die Rolle enthält eine Berechtigungsrichtlinie, die Lesezugriff auf den angegebenen S3-Bucket gewährt. Sie enthält auch eine Vertrauensrichtlinie, die es der Amazon-EC2-Instance ermöglicht, die Rolle zu übernehmen und die temporären Anmeldeinformationen abzurufen. Wenn die Anwendung auf der Instance ausgeführt wird, kann sie mithilfe der temporären Anmeldeinformationen der Rolle auf den photos-Bucket zugreifen. Der Administrator muss dem Entwickler keine Berechtigung für den Zugriff auf den photos-Bucket gewähren und der Entwickler muss die Anmeldeinformationen nie weitergeben oder verwalten.



1. Der Administrator erstellt mit IAM die Rolle **`Get-pics`**. In der Vertrauensrichtlinie der Rolle legt der Administrator fest, dass nur Amazon-EC2-Instances die Rolle übernehmen können. In der Berechtigungsrichtlinie der Rolle gibt der Administrator Leseberechtigungen für den Bucket `photos` an.
2. Ein Entwickler startet eine Amazon-EC2-Instance und weist die Rolle `Get-pics` dieser Instance zu.

> **Note**
>
> Wenn Sie die IAM-Konsole verwenden, wird das Instance-Profil für Sie verwaltet und ist Ihnen gegenüber weitestgehend transparent. Wenn Sie jedoch die API AWS CLI oder verwenden, um die Rolle und die Amazon EC2 EC2-Instance zu erstellen und

zu verwalten, müssen Sie das Instance-Profil erstellen und ihm die Rolle in separaten Schritten zuweisen. In diesem Fall müssen Sie den Instance-Profilnamen anstelle des Rollennamens angeben, wenn Sie die Instance starten.

3. Wenn die Anwendung ausgeführt wird, erhält sie temporäre Sicherheitsanmeldeinformationen von den Amazon EC2-Instance-Metadaten, wie unter Abrufen von Sicherheitsanmeldeinformationen von Instance-Metadaten beschrieben. Hierbei handelt es sich um temporäre Sicherheitsanmeldeinformationen für die Rolle, die für einen begrenzten Zeitraum gültig sind.

   Bei einigen AWS SDKs kann der Entwickler einen Anbieter verwenden, der die temporären Anmeldeinformationen transparent verwaltet. (In der Dokumentation für die einzelnen AWS SDKs werden die Funktionen beschrieben, die von diesem SDK für die Verwaltung von Anmeldeinformationen unterstützt werden.)

   Alternativ kann die Anwendung die temporären Anmeldeinformationen auch direkt aus den Instance-Metadaten der Amazon-EC2-Instance beziehen. Die Anmeldeinformationen und die zugehörigen Werte sind in der Kategorie `iam/security-credentials/`*`role-name`* (in diesem Fall `iam/security-credentials/Get-pics`) der Metadaten verfügbar. Wenn die Anwendung die Anmeldeinformationen aus den Instance-Metadaten bezieht, können die Anmeldeinformationen zwischengespeichert werden.

4. Mithilfe der erhaltenen temporären Anmeldeinformationen greift die Anwendung auf den Bucket "photos" zu. Aufgrund der zur Rolle **`Get-pics`** angefügten Richtlinie verfügt die Anwendung nur über Leseberechtigungen.

   Die in der Instance verfügbaren temporären Sicherheitsanmeldeinformationen werden vor deren Ablauf automatisch aktualisiert, sodass immer gültige Anmeldeinformationen verfügbar sind. Die Anwendung muss lediglich sicherstellen, dass sie neue Anmeldeinformationen von den Instance-Metadaten erhält, bevor die aktuellen Anmeldeinformationen ablaufen. Es ist möglich, das AWS SDK zur Verwaltung von Anmeldeinformationen zu verwenden, sodass die Anwendung keine zusätzliche Logik zum Aktualisieren der Anmeldeinformationen enthalten muss. Zum Beispiel das Instanceiieren von Clients mit Anbietern von Anmeldeinformationen für Instance-Profile. Erhält jedoch die Anwendung temporäre Anmeldeinformationen von den Instance-Metadaten und speichert diese im Cache, sollte die Anwendung die Anmeldeinformationen stündlich oder mindestens 15 Minuten vor deren Ablauf aktualisieren. Die Ablaufzeit ist in den Informationen enthalten, die in der Kategorie `iam/security-credentials/`*`role-name`* zurückgegeben wird.

Erforderliche Berechtigungen für die Nutzung von Rollen mit Amazon EC2

Um eine Instance mit einer Rolle zu starten, muss der Entwickler über die Berechtigung zum Starten von Amazon-EC2-Instances und die Berechtigung zum Übergeben von IAM-Rollen verfügen.

Mit der folgenden Beispielrichtlinie können Benutzer die verwenden AWS Management Console , um eine Instanz mit einer Rolle zu starten. Die Richtlinie enthält Platzhalter (*), um einem Benutzer die Übergabe von beliebigen Rollen und die Ausführung beliebiger Amazon-EC2-Aktionen zu gestatten. Die `ListInstanceProfiles`-Aktion ermöglicht den Benutzern, alle im AWS-Konto verfügbaren Rollen anzuzeigen.

Example Beispielrichtlinie, mit der eine Benutzerberechtigung zum Starten einer Instance mit einer beliebigen Rolle über die Amazon EC2-Konsole erteilt wird

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

Einschränken, welche Rollen an Amazon EC2 EC2-Instances übergeben werden können (mit) PassRole

Sie können die `PassRole`-Berechtigung verwenden, um einzuschränken, welche Rolle ein Benutzer an eine Amazon-EC2-Instance übergeben kann, wenn der Benutzer die Instance startet. Dadurch können Sie verhindern, dass ein Benutzer Anwendungen ausführt, die über mehr Berechtigungen als der Benutzer verfügen, das heißt, um zu vermeiden, dass der Benutzer erhöhte Berechtigungen erhält. Angenommen, die Benutzerin Alice verfügt nur über Berechtigungen zum Starten von Amazon-EC2-Instances und für den Zugriff auf Amazon-S3-Buckets, aber die von ihr an eine Amazon-EC2-Instance übergeben Rolle verfügt über Zugriffsberechtigungen auf IAM und Amazon DynamoDB. In diesem Fall könnte Alice in der Lage sein, die Instance zu starten, sich bei ihr anzumelden, temporäre Sicherheitsanmeldeinformationen zu erhalten und dann IAM- oder DynamoDB-Aktionen durchzuführen, für die sie nicht autorisiert ist.

Um einzuschränken, welche Rollen ein Benutzer an eine Amazon-EC2-Instance übergeben kann, erstellen Sie eine Richtlinie, die die Aktion `PassRole` zulässt. Anschließend fügen Sie die Richtlinie dem Benutzer (oder einer IAM-Gruppe, zu der der Benutzer gehört) zu, der Amazon-EC2-Instances starten wird. Im Element `Resource` der Richtlinie führen Sie die Rolle oder Rollen auf, die der Benutzer an Amazon-EC2-Instances übergeben darf. Wenn der Benutzer eine Instance startet und ihr ein Rolle zuordnet, prüft Amazon EC2, ob der Benutzer zum Übergeben dieser Rolle berechtigt ist. Selbstverständlich sollten Sie auch sicherstellen, dass die Rolle, die der Benutzer übergeben darf, nicht mehr Berechtigungen enthält, als für den Benutzer vorgesehen ist.

> **Note**
>
> `PassRole` ist keine API-Aktion im Sinne von `RunInstances` oder `ListInstanceProfiles`. Stattdessen handelt es sich um eine Berechtigung, die AWS überprüft, wann immer ein Rollen-ARN als Parameter an eine API übergeben wird (oder die Konsole dies im Namen des Benutzers tut). Damit kann der Administrator steuern, welche Rollen von welchen Benutzern übergeben werden dürfen. In diesem Fall wird sichergestellt, dass der Benutzer eine spezifische Rolle einer Amazon EC2-Instance anfügen darf.

Example Beispielrichtlinie, mit der eine Benutzerberechtigung zum Starten einer Amazon-EC2-Instance mit einer bestimmten Rolle erteilt wird

In der folgenden Beispielrichtlinie wird den Benutzern ermöglicht, die Amazon EC2-API zum Starten einer Instance mit einer Rolle zu verwenden. Das Element `Resource` gibt den Amazon-

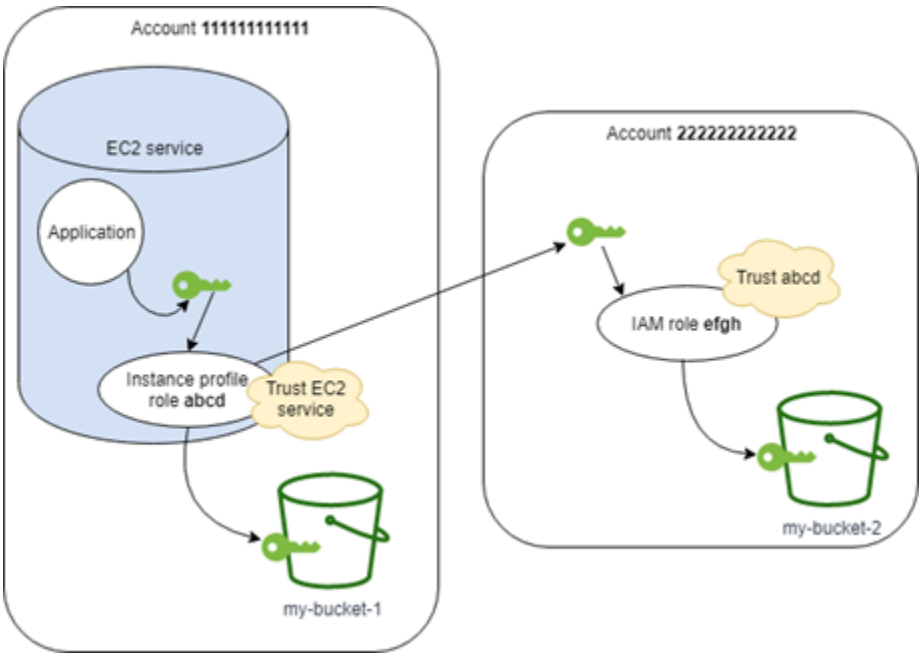
Ressourcenamen (ARN) einer Rolle an. Indem Sie den ARN angeben, erteilt die Richtlinie dem Benutzer die Berechtigung, nur die Rolle `Get-pics` zu übergeben. Wenn der Benutzer versucht, eine andere Rolle beim Starten einer Instance anzugeben, schlägt die Aktion fehl. Der Benutzer ist nicht zum Ausführen einer beliebigen Instance berechtigt, unabhängig davon, ob er eine Rolle übergibt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id:role/Get-pics"
    }
  ]
}
```

Einer Instance-Profilrolle den Wechsel zu einer Rolle in eine anderen Konto erlauben

Sie können einer Anwendung, die auf einer Amazon EC2-Instance ausgeführt wird, das Ausführen von Befehlen in einem anderen Konto erlauben. Dazu müssen Sie der Amazon-EC2-Instance-Rolle im ersten Konto den Wechsel zu einer Rolle im zweiten Konto erlauben.

Stellen Sie sich vor, Sie verwenden zwei AWS-Konten und möchten einer Anwendung, die auf einer Amazon EC2 EC2-Instance ausgeführt wird, erlauben, AWS CLIBefehle in beiden Konten auszuführen. Gehen Sie davon aus, dass die Amazon-EC2-Instance im Konto `111111111111` vorhanden ist. Diese Instance enthält die `abcd`-Instance-Profilrolle, die der Anwendung erlaubt, schreibgeschützte Amazon S3-Aufgaben für den `my-bucket-1`-Bucket innerhalb desselben `111111111111` Kontos auszuführen. Die Anwendung muss jedoch auch die Möglichkeit haben, die `efgh`-kontoübergreifende Rolle anzunehmen, um auf den Amazon S3-Bucket `my-bucket-2` im Konto `222222222222` zuzugreifen.



Der Instance-Profil-Rolle `abcd` von Amazon EC2 muss die folgende Berechtigungsrichtlinie zugeordnet sein, damit die Anwendung auf den `my-bucket-1` Amazon-S3-Bucket zugreifen kann:

Konto 111111111111 ***abcd*** Rollen-Berechtigungsrichtlinie

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Die `abcd`-Rolle muss dem Amazon EC2-Service vertrauen, um die Rolle anzunehmen. Dazu benötigt die `abcd`-Rolle die folgende Vertrauensrichtlinie:

Konto 111111111111 ***abcd*** Rolle Treuhandpolitik

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}
```

Gehen Sie davon aus, dass die kontoübergreifende `efgh`-Rolle die Ausführung schreibgeschützter Amazon S3-Aufgaben für den `my-bucket-2`-Bucket im selben `222222222222`-Konto zulässt. Dazu muss der kontoübergreifenden `efgh`-Rolle die folgende Berechtigungsrichtlinie zugeordnet sein:

Konto 222222222222 ***efgh*** Rollen-Berechtigungsrichtlinie

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

Die `efgh`-Rolle muss für die Übernahme der `abcd`-Instance-Profilrolle vertrauen. Dazu benötigt die `efgh`-Rolle die folgende Vertrauensrichtlinie:

Konto 222222222222 ***`efgh`*** Rolle Treuhandpolitik

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Was sind die ersten Schritte?

Um die Funktionsweise einer Rolle mit Amazon-EC2-Instances nachzuvollziehen, müssen Sie mithilfe der IAM-Konsole eine Rolle erstellen, eine Amazon-EC2-Instance starten, die diese Rolle verwendet und dann die laufende Instance analysieren. Sie können die Instance-Metadaten untersuchen, um nachzuvollziehen, wie die Instance die temporären Anmeldeinformationen der Rolle erhält. Sie können außerdem nachvollziehen, wie eine auf der Instance ausgeführte Anwendung diese Rolle verwenden kann. Verwenden Sie die folgenden Ressourcen, um weitere Informationen zu erhalten.

-
- SDK-Anleitungen. Die AWS SDK-Dokumentation enthält exemplarische Vorgehensweisen, die eine Anwendung zeigen, die auf einer Amazon EC2 EC2-Instance ausgeführt wird und temporäre Anmeldeinformationen für Rollen verwendet, um einen Amazon S3 S3-Bucket zu lesen. Jede der folgenden Anleitungen beschreibt die gleiche Vorgehensweise in einer anderen Programmiersprache:
  - Konfigurieren von IAM-Rollen Amazon EC2 mit dem SDK for Java im AWS SDK for Java Developer Guide
  - Starten einer Amazon-EC2-Instance mit dem SDK für .NET im AWS SDK for .NET - Entwicklerhandbuch
  - Erstellen einer Amazon EC2 Instance mit dem SDK for Ruby im AWS SDK for Ruby Developer Guide

Ähnliche Informationen

Weitere Informationen zum Erstellen von Rollen für Amazon-EC2-Instances finden Sie in den folgenden Informationen:

- Weitere Informationen zur Verwendung von IAM-Rollen mit Amazon EC2 EC2-Instances finden Sie im Amazon EC2 EC2-Benutzerhandbuch.
- Informationen zum Erstellen einer Rolle finden Sie unter Erstellen von IAM-Rollen.
- Weitere Informationen zum Verwenden von temporären Sicherheitsanmeldeinformationen finden Sie unter Temporäre IAM Sicherheitsanmeldeinformationen.
- Wenn Sie mit der IAM-API oder der Befehlszeilenschnittstelle (CLI) arbeiten möchten, müssen Sie IAM-Instance-Profile erstellen und verwalten. Weitere Informationen zu Instance-Profilen finden Sie unter Verwenden von Instance-Profilen.

- Weitere Informationen zu temporären Sicherheitsanmeldedaten für Rollen in den Instance-Metadaten finden Sie unter Abrufen von Sicherheitsanmeldedaten aus Instance-Metadaten im Amazon EC2 EC2-Benutzerhandbuch.

Verwenden von Instance-Profilen

Verwenden Sie ein Instance-Profil, um eine IAM-Rolle an eine EC2-Instance zu übergeben. Weitere Informationen finden Sie unter IAM-Rollen für Amazon EC2 im Amazon EC2 EC2-Benutzerhandbuch.

Verwalten von Instance-Profilen (Konsole)

Wenn Sie die verwenden AWS Management Console , um eine Rolle für Amazon EC2 zu erstellen, erstellt die Konsole automatisch ein Instance-Profil und weist diesem den gleichen Namen wie die Rolle zu. Wenn Sie dann die Amazon EC2-Konsole verwenden, um eine Instance mit einer IAM-Rolle zu starten, können Sie eine Rolle auswählen, die der Instance zugewiesen wird. In der Konsole ist die Liste, die angezeigt wird, tatsächlich eine Liste von Instance-Profilnamen. Die Konsole erstellt kein Instance-Profil für eine Rolle, die nicht mit Amazon EC2 verknüpft ist.

Sie können die verwenden AWS Management Console , um IAM-Rollen und Instance-Profile für Amazon EC2 zu löschen, wenn die Rolle und das Instance-Profil denselben Namen haben. Weitere Informationen zum Löschen von Instanceprofilen finden Sie unter Löschen von Rollen oder Instance-Profilen.

Verwaltung von Instanzprofilen (AWS CLI oder AWS API)

Wenn Sie Ihre Rollen über die AWS CLI oder die AWS API verwalten, erstellen Sie Rollen und Instanzprofile als separate Aktionen. Da Rollen und Instance-Profile unterschiedliche Namen haben können, müssen Sie die Namen der Instance-Profile sowie die Namen der Rollen kennen, die sie enthalten. Auf diese Weise können Sie das richtige Instance-Profil wählen, wenn Sie eine EC2 Instance starten.

Sie können Ihren IAM-Ressourcen Tags anfügen, einschließlich Instance-Profile, um den Zugriff auf sie zu identifizieren, zu organisieren und zu kontrollieren. Sie können Instanzprofile nur taggen, wenn Sie die AWS API AWS CLI oder verwenden.

> **Note**
>
> Ein Instance-Profil kann nur eine einzige IAM-Rolle enthalten, eine Rolle kann jedoch in mehreren Instance-Profile enthalten sein. Diese Beschränkung auf eine Rolle pro Instance-

Profil kann nicht erhöht werden. Sie können die vorhandene Rolle entfernen und dann eine andere Rolle zu einem Instance-Profil hinzufügen. AWS Aus Gründen der Konsistenz müssen Sie dann warten, bis die Änderung für alle sichtbar ist. Wenn Sie die Änderung erzwingen möchten, müssen Sie die Zuweisung des Instance-Profils aufheben und dann das Instance-Profil zuweisen, oder Sie beenden Ihre Instance und starten sie neu.

Verwalten von Instance-Profilen (AWS CLI)

Sie können die folgenden AWS CLI Befehle verwenden, um mit Instanzprofilen in einem AWS Konto zu arbeiten.

- Erstellen eines Instance-Profils: `aws iam create-instance-profile`
- Markieren eines Instance-Profils: `aws iam tag-instance-profile`
- Auflisten von Tags für ein Instance-Profil: `aws iam list-instance-profile-tags`
- Entfernen der Markierung eines Instance-Profils: `aws iam untag-instance-profile`
- Hinzufügen einer Rolle zu einem Instance-Profil: `aws iam add-role-to-instance-profile`
- Auflisten von Instance-Profilen: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Abrufen von Informationen zu einem Instance-Profil: `aws iam get-instance-profile`
- Entfernen einer Rolle aus einem Instance-Profil: `aws iam remove-role-from-instance-profile`
- Löschen eines Instance-Profils: `aws iam delete-instance-profile`

Mithilfe der folgenden Befehle können Sie eine Rolle auch einer bereits ausgeführten EC2-Instance anfügen. Weitere Informationen finden Sie unter IAM-Rollen für Amazon EC2.

- Anfügen eines Instance-Profils mit einer Rolle an eine angehaltene oder ausgeführte EC2-Instance: `aws ec2 associate-iam-instance-profile`
- Abrufen von Informationen zu einem Instance-Profil, das an eine EC2-Instance angefügt ist: `aws ec2 describe-iam-instance-profile-associations`
- Trennen eines Instance-Profils mit einer Rolle von einer angehaltenen oder ausgeführten EC2-Instance: `aws ec2 disassociate-iam-instance-profile`

Verwalten von Instance-Profilen (AWS -API)

Sie können die folgenden AWS API-Operationen aufrufen, um mit Instanzprofilen in einem zu arbeiten AWS-Konto.

- Erstellen eines Instance-Profils: `CreateInstanceProfile`
- Markieren eines Instance-Profils: `TagInstanceProfile`
- Auflisten von Tags auf einem Instance-Profil: `ListInstanceProfileTags`
- Entfernen der Markierung eines Instance-Profils: `UntagInstanceProfile`
- Hinzufügen einer Rolle zu einem Instance-Profil: `AddRoleToInstanceProfile`
- Auflisten von Instance-Profilen: `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Abrufen von Informationen zu einem Instance-Profil: `GetInstanceProfile`
- Entfernen einer Rolle aus einem Instance-Profil: `RemoveRoleFromInstanceProfile`
- Löschen eines Instance-Profils: `DeleteInstanceProfile`

Außerdem können Sie eine Rolle einer bereits ausgeführten EC2 Instance anfügen, indem Sie die folgenden Operationen aufrufen. Weitere Informationen finden Sie unter IAM-Rollen für Amazon EC2.

- Anfügen eines Instance-Profils mit einer Rolle an eine angehaltene oder ausgeführte EC2-Instance: `AssociateIamInstanceProfile`
- Abrufen von Informationen zu einem Instance-Profil, das an eine EC2-Instance angefügt ist: `DescribeIamInstanceProfileAssociations`
- Trennen eines Instance-Profils mit einer Rolle von einer angehaltenen oder ausgeführten EC2-Instance: `DisassociateIamInstanceProfile`

Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen

> **Warning**
>
> Wenn Sie die Schritte auf dieser Seite befolgen, wird allen Benutzern mit aktuellen Sitzungen, die durch Übernahme der Rolle erstellt wurden, der Zugriff auf alle AWS Aktionen und Ressourcen verweigert. Dies kann dazu führen, dass Benutzer nicht gespeicherte Arbeit verlieren.

Wenn Sie Benutzern den Zugriff auf die AWS Management Console mit einer langen Sitzungsdauer (z. B. 12 Stunden) gestatten, laufen ihre temporären Anmeldeinformationen nicht so schnell ab. Wenn Benutzer versehentlich ihre Anmeldeinformationen nicht autorisierten Dritten preisgeben, haben diese für die Dauer der Sitzung entsprechenden Zugriff. Sie können jedoch sofort alle Berechtigungen für die Anmeldeinformationen der Rolle widerrufen, die vor einem bestimmten Zeitpunkt ausgestellt wurden, wenn dies erforderlich ist. Alle temporären Anmeldeinformationen für diese Rolle, die vor dem angegebenen Zeitpunkt ausgestellt wurden, werden ungültig. Dies zwingt alle Benutzer, sich erneut zu authentifizieren und neue Anmeldeinformationen anfordern.

> **Note**
>
> Die Sitzung für eine serviceverknüpfte Rolle können Sie nicht widerrufen.

Wenn Sie mithilfe des in diesem Thema beschriebenen Verfahrens Berechtigungen für eine Rolle entziehen, AWS fügt der Rolle eine neue Inline-Richtlinie hinzu, die alle Berechtigungen für alle Aktionen verweigert. Sie enthält eine Bedingung, die die Einschränkungen nur dann anwendet, wenn der Benutzer die Rolle vor dem Zeitpunkt übernommen hat, zu dem Sie die Berechtigungen widerrufen. Wenn der Benutzer die Rolle übernimmt, nachdem Sie die Berechtigungen entzogen haben, dann gilt die Zugriffsverweigerungsrichtlinie nicht für diesen Benutzer.

Weitere Informationen zum Verweigern des Zugriffs finden Sie unter Deaktivieren von Berechtigungen für temporäre Sicherheitsanmeldeinformationen.

> **Important**
>
> Diese Zugriffsverweigerungsrichtlinie gilt für alle Benutzer der angegebenen Rolle, nicht nur für diejenigen mit Konsolensitzungen mit längerer Dauer.

Mindestberechtigungen für das Entziehen von Sitzungsberechtigungen einer Rolle

Um einer Rolle Sitzungsberechtigungen erfolgreich zu entziehen, müssen Sie über die `PutRolePolicy`-Berechtigung für die Rolle verfügen. Auf diese Weise können Sie der Rolle die Inline-Richtlinie `AWSRevokeOlderSessions` anfügen.

Widerrufen von Sitzungsberechtigungen

Sie können einer Rolle die Sitzungsberechtigungen entziehen, um allen Benutzern, die die Rolle übernommen haben, alle Berechtigungen zu verweigern.

> **Note**
>
> Sie können in IAM keine Rollen bearbeiten, die anhand von IAM Identity Center-Berechtigungssätzen erstellt wurden. Sie müssen die aktive Sitzung mit dem Berechtigungssatz für einen Benutzer in IAM Identity Center widerrufen. Weitere Informationen finden Sie unter Widerrufen aktiver IAM-Rollensitzungen, die mit Berechtigungssätzen erstellt wurden, im IAM Identity Center-Benutzerhandbuch.

So verweigern Sie sofort sämtliche Berechtigungen für alle aktuellen Benutzer der Anmeldeinformationen der Rolle

1. Melden Sie sich unter https://console.aws.amazon.com/iam/ bei der IAM-Konsole an AWS Management Console und öffnen Sie sie.
2. Wählen Sie im Navigationsbereich Rollen und dann den Namen (nicht das Kontrollkästchen) der Rolle, deren Berechtigungen Sie entziehen möchten.
3. Klicken Sie auf der Seite Summary (Übersicht) für die ausgewählte Rolle auf die Registerkarte Revoke sessions (Sitzungen widerrufen).
4. Wählen Sie auf der Registerkarte Revoke sessions (Sitzungen widerrufen) die Option Revoke active sessions (Aktive Sitzungen widerrufen).
5. AWS fordert Sie auf, die Aktion zu bestätigen. Wählen Sie das Kontrollkästchen I acknowledge that I am revoking all active sessions for this role. (Ich bestätige, dass ich alle aktiven Sitzungen für diese Rolle widerrufe.) aus und dann im Dialogfeld Revoke active sessions (Aktive Sitzungen widerrufen).

   IAM fügt dann der Rolle eine Richtlinie mit dem Namen `AWSRevokeOlderSessions` hinzu. Nachdem Sie Aktive Sitzungen widerrufen ausgewählt haben, verweigert die Richtlinie Benutzern, die diese Rolle in der Vergangenheit übernommen haben, sowie etwa 30 Sekunden in der future jeglichen Zugriff. Bei dieser Wahl des future Zeitpunkts wird die Verbreitungsverzögerung der Richtlinie berücksichtigt, um eine neue Sitzung zu bearbeiten, die erworben oder verlängert wurde, bevor die aktualisierte Richtlinie in einer bestimmten Region in Kraft trat. Jeder Benutzer, der die Rolle mehr als etwa 30 Sekunden, nachdem Sie Aktive

Sitzungen widerrufen ausgewählt haben, übernimmt, ist davon nicht betroffen. Um zu erfahren, warum Änderungen nicht immer sofort sichtbar sind, siehe Änderungen, die ich vornehme, sind nicht immer direkt sichtbar.

> **Note**
>
> Wenn Sie aktive Sitzungen später erneut widerrufen, werden der Datums- und Zeitstempel in der Richtlinie aktualisiert, und es werden erneut allen Benutzern, die die Rolle vor dem neuen angegebenen Zeitpunkt übernommen haben, alle Berechtigungen verweigert.

Gültige Benutzer, deren Sitzungen auf diese Weise aufgehoben wurden, benötigen temporäre Anmeldeinformationen für eine neue Sitzung, um weiterarbeiten zu können. Die AWS CLI speichert Anmeldeinformationen, bis sie ablaufen. Um die CLI zu zwingen, zwischengespeicherte Anmeldeinformationen, die nicht mehr gültig sind, zu löschen und zu aktualisieren, führen Sie einen der folgenden Befehle aus:

Linux, macOS oder Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Widerrufen von Sitzungsberechtigungen vor einem bestimmten Zeitpunkt

Sie können Sitzungsberechtigungen auch jederzeit Ihrer Wahl widerrufen, indem Sie das SDK AWS CLI oder das SDK verwenden, um einen Wert für den `aws: TokenIssue Zeit` Schlüssel im Condition-Element einer Richtlinie anzugeben.

Diese Richtlinie verweigert alle Berechtigungen, wenn der Wert von `aws:TokenIssueTime` vor dem angegebenen Datum und der angegebenen Uhrzeit liegt. Der Wert von `aws:TokenIssueTime` entspricht der präzisen Uhrzeit, zu der die temporären Sicherheitsanmeldeinformationen erstellt wurden. Der `aws:TokenIssueTime` Wert ist nur im Kontext von AWS Anfragen vorhanden, die mit temporären Sicherheitsanmeldedaten signiert wurden. Daher wirkt sich die Deny-Anweisung in der Richtlinie nicht auf Anfragen aus, die mit den langfristigen Anmeldeinformationen des IAM-Benutzers signiert wurden.

Diese Richtlinie kann auch einer Rolle zugeordnet werden. In diesem Fall wirkt sich die Richtlinie nur auf die temporären Sicherheitsanmeldeinformationen aus, die vor dem angegebenen Datum und der angegebenen Uhrzeit von der Rolle erstellt wurden.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Gültige Benutzer, deren Sitzungen auf diese Weise aufgehoben wurden, benötigen temporäre Anmeldeinformationen für eine neue Sitzung, um weiterarbeiten zu können. Die Anmeldeinformationen werden AWS CLI zwischengespeichert, bis sie ablaufen. Um die CLI zu zwingen, zwischengespeicherte Anmeldeinformationen, die nicht mehr gültig sind, zu löschen und zu aktualisieren, führen Sie einen der folgenden Befehle aus:

Linux, macOS oder Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Verwalten von IAM-Rollen

Gelegentlich müssen Sie die von Ihnen erstellten Rollen ändern oder löschen. Zum Ändern einer Rolle können Sie die folgenden Schritte ausführen:

- Ändern der Richtlinien, die mit der Rolle verbunden sind
- Ändern, wer auf die Rolle zugreifen kann
- Bearbeiten der Berechtigungen, die den Benutzern von der Rolle erteilt werden

- Ändern Sie die Einstellung für die maximale Sitzungsdauer für Rollen, von denen angenommen wird AWS Management Console, dass sie die API AWS CLI oder verwenden

Sie können auch nicht mehr benötigte Rollen löschen. Sie können Ihre Rollen über die AWS Management Console AWS CLI, und die API verwalten.

Themen

- Ändern einer Rolle
- Löschen von Rollen oder Instance-Profilen

Ändern einer Rolle

Sie können die AWS Management Console AWS CLI, oder die IAM-API verwenden, um Änderungen an einer Rolle vorzunehmen.

Themen

- Anzeigen des Rollenzugriffs
- Generieren einer Richtlinie basierend auf Zugriffsinformationen
- Ändern einer Rolle (Konsole)
- Ändern einer Rolle (AWS CLI)
- Eine Rolle ändern (AWS API)

Anzeigen des Rollenzugriffs

Bevor Sie die Berechtigungen für eine Rolle ändern, sollten Sie ihre kürzliche Service-Level-Aktivität überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Generieren einer Richtlinie basierend auf Zugriffsinformationen

Manchmal können Sie einer IAM-Entität (Benutzer oder Rolle) Berechtigungen erteilen, die über das hinausgehen, was diese benötigen. Um Ihnen beim Verfeinern der Berechtigungen zu helfen, können Sie eine IAM-Richtlinie generieren, die auf der Zugriffsaktivität für eine Entity basiert. IAM Access Analyzer überprüft Ihre AWS CloudTrail Protokolle und generiert eine Richtlinienvorlage,

die die Berechtigungen enthält, die von der Entität in dem von Ihnen angegebenen Zeitraum verwendet wurden. Sie können die Vorlage verwenden, um eine verwaltete Richtlinie mit definierten Berechtigungen zu erstellen und sie dann an die IAM-Rolle anzuhängen. Auf diese Weise gewähren Sie nur die Berechtigungen, die der Benutzer oder die Rolle benötigt, um mit AWS Ressourcen für Ihren speziellen Anwendungsfall zu interagieren. Weitere Informationen hierzu finden Sie unter Generieren von Richtlinien basierend auf Zugriffsaktivitäten.

Ändern einer Rolle (Konsole)

Sie können den verwenden AWS Management Console , um eine Rolle zu ändern. Informationen zum Ändern der Gruppe von Tags einer Rolle finden Sie unter Verwalten von Tags auf IAM-Rollen (Konsole).

Themen

- Ändern einer Rollenvertrauensrichtlinie (Konsole)
- Ändern einer Rollenberechtigungsrichtlinie (Konsole)
- Ändern einer Rollenbeschreibung (Konsole)
- Ändern der maximalen Sitzungsdauer einer Rolle (Konsole)
- Ändern einer Rollenberechtigungsgrenze (Konsole)

Ändern einer Rollenvertrauensrichtlinie (Konsole)

Um zu ändern, wer eine Rolle übernehmen kann, müssen Sie die Vertrauensrichtlinie der Rolle bearbeiten. Sie können die Vertrauensrichtlinie für eine serviceverknüpfte Rolle nicht ändern.

Hinweise

- Wenn ein Benutzer als Auftraggeber in der Vertrauensrichtlinie einer Rolle aufgeführt ist, aber die Rolle nicht übernehmen kann, überprüfen Sie die Berechtigungsgrenze des Benutzers. Wenn eine Berechtigungsgrenze für den Benutzer festgelegt ist, dann muss sie die `sts:AssumeRole`-Aktion zulassen.
- Damit Benutzer die aktuelle Rolle innerhalb einer Rollensitzung wieder übernehmen können, geben Sie den Rollen-ARN oder AWS-Konto ARN als Principal in der Rollenvertrauensrichtlinie an. AWS-Services die Rechenressourcen wie Amazon EC2, Amazon ECS, Amazon EKS und Lambda bereitstellen, stellen temporäre Anmeldeinformationen bereit und aktualisieren diese Anmeldeinformationen automatisch. Dadurch wird sichergestellt, dass Sie immer über gültige Anmeldeinformationen verfügen.

Für diese Dienste ist es nicht erforderlich, die aktuelle Rolle erneut anzunehmen, um temporäre Anmeldeinformationen zu erhalten. Wenn Sie jedoch beabsichtigen, Sitzungs-Tags oder eine Sitzungsrichtlinie zu übergeben, müssen Sie die aktuelle Rolle erneut annehmen.

So ändern Sie eine Rollenvertrauensrichtlinie (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. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus.
3. Wählen Sie in der Rollenliste in Ihrem Konto den Namen der zu ändernden Rolle.
4. Wählen Sie die Registerkarte Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
5. Bearbeiten Sie die Vertrauensrichtlinie nach Bedarf. Um weitere Auftraggeber hinzuzufügen, die die Rolle übernehmen können, geben Sie sie im `Principal`-Element an. Der folgende Richtlinienausschnitt zeigt beispielsweise, wie AWS-Konten in dem Element auf zwei verwiesen wird: `Principal`

   ```
   "Principal": {
     "AWS": [
       "arn:aws:iam::111122223333:root",
       "arn:aws:iam::444455556666:root"
     ]
   },
   ```

   Wenn Sie einen Auftraggeber in einem anderen Konto angeben, ist das Hinzufügen eines Kontos zur Vertrauensrichtlinie einer Rolle nur die Hälfte der Einrichtung einer kontoübergreifenden Vertrauensbeziehung. Standardmäßig kann kein Benutzer in den vertrauenswürdigen Konten die Rolle übernehmen. Der Administrator für das neue vertrauenswürdige Konto muss den Benutzern die Berechtigung erteilen, die Rolle zu übernehmen. Dazu muss der Administrator eine Richtlinie erstellen oder bearbeiten, die an den Benutzer angehängt ist, um dem Benutzer den Zugriff auf die Aktion `sts:AssumeRole` zu ermöglichen. Weitere Informationen finden Sie im folgenden Verfahren oder unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

Der folgende Richtlinienausschnitt zeigt, wie auf zwei AWS Dienste im Element verwiesen wird: `Principal`

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. Wenn Sie die Bearbeitung Ihrer Vertrauensrichtlinie abgeschlossen haben, wählen Sie Update policy (Richtlinie aktualisieren), um Ihre Änderungen zu speichern.

   Weitere Informationen über die Richtlinienstruktur und -syntax finden Sie unter Berechtigungen und Richtlinien in IAM und IAM-JSON-Richtlinienelementreferenz.

So erteilen Sie Benutzern in einem vertrauenswürdigen Konto die Berechtigung zur Verwendung der Rolle (Konsole)

Weitere Informationen und Details zu dieser Vorgehensweise finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

1. Melden Sie sich bei dem vertrauenswürdigen externen Anbieter an. AWS-Konto
2. Legen Sie fest, ob die Berechtigungen einem Benutzer oder einer Gruppe angefügt werden. Wählen Sie im Navigationsbereich der IAM-Konsole entsprechend Users (Benutzer) oder Groups (Gruppen).
3. Wählen Sie den Namen des Benutzers oder der Gruppe aus, der Sie die Zugriffsberechtigung erteilen möchten, und klicken Sie auf die Registerkarte Permissions (Berechtigungen).
4. Führen Sie eine der folgenden Aktionen aus:

   - Um eine vom Kunden verwaltete Richtlinien zu bearbeiten, wählen Sie den Namen der Richtlinie, Edit policy (Richtlinie bearbeiten) und die Registerkarte JSON. Sie können eine AWS verwaltete Richtlinie nicht bearbeiten. AWS Verwaltete Richtlinien werden mit dem AWS Symbol () angezeigt. Weitere Informationen zum Unterschied zwischen AWS verwalteten Richtlinien

und kundenverwalteten Richtlinien finden Sie unterVerwaltete Richtlinien und eingebundene Richtlinien.

- Um eine Inlinerichtlinie zu bearbeiten, wählen Sie den Pfeil neben dem Namen der Richtlinie und dann Edit policy (Richtlinie bearbeiten).

5. Fügen Sie im Richtlinien-Editor ein neues `Statement`-Element hinzu, welches folgende Angaben macht:

```
{
  "Effect": "Allow",
  "Action": "sts:AssumeRole",
  "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
}
```

   Ersetzen Sie den ARN in der Anweisung durch den ARN der Rolle, die der Benutzer übernehmen kann.

6. Folgen Sie den Anweisungen auf dem Bildschirm, um die Bearbeitung der Richtlinie abzuschließen.

Ändern einer Rollenberechtigungsrichtlinie (Konsole)

Um die von der Rolle zugelassenen Berechtigungen zu ändern, modifizieren Sie die Berechtigungsrichtlinie (oder Berechtigungsrichtlinien) der Rolle. Sie können die Berechtigungsrichtlinie für eine serviceverknüpfte Rolle in IAM nicht ändern. Möglicherweise können Sie die Berechtigungsrichtlinie innerhalb des Service ändern, der von der Rolle abhängt. Um zu überprüfen, ob ein Service dieses Feature unterstützt, lesen Sie AWS Dienste, die mit IAM funktionieren und suchen Sie nach den Services mit Yes (Ja) in der Spalte Service-linked roles (Serviceverknüpfte Rollen). Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

So ändern Sie die Berechtigungen einer Rolle (Konsole)

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus.
3. Wählen Sie den Namen der zu ändernden Rolle aus und klicken Sie anschließend auf die Registerkarte Permissions (Berechtigungen).
4. Führen Sie eine der folgenden Aktionen aus:

- Um eine vorhandene vom Kunden verwaltete Richtlinien zu bearbeiten, wählen Sie den Namen der Richtlinie und dann Edit policy (Richtline bearbeiten).

> **Note**
>
> Eine AWS verwaltete Richtlinie kann nicht bearbeitet werden. AWS Verwaltete Richtlinien werden mit dem AWS Symbol () angezeigt. Weitere Informationen zum Unterschied zwischen von AWS und vom Kunden verwaltete Richtlinien finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien.

- Um Rolle eine vorhandene verwaltete Richtlinie an die Rolle anzufügen, wählen Sie Add permissions (Berechtigungen hinzufügen) und dann Attach policies (Richtlinien anfügen).
- Um eine bestehende Inline-Richtlinie zu bearbeiten, erweitern Sie die Richtlinie und wählen Sie Edit (Bearbeiten).
- Um eine neue Inline-Richtlinie einzubetten, wählen Sie Add permissions (Berechtigungen hinzufügen) und dann Create inline policy (Inline-Richtlinie erstellen).
- Um eine bestehende Richtlinie aus der Rolle zu entfernen, aktivieren Sie das Kontrollkästchen neben dem Richtliniennamen und wählen Sie dann Entfernen aus.

Ändern einer Rollenbeschreibung (Konsole)

Um die Beschreibung einer Rolle zu ändern, modifizieren Sie den Text der Beschreibung.

So ändern Sie die Beschreibung einer Rolle (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. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus.
3. Wählen Sie den Namen der zu ändernden Rolle.
4. Wählen Sie im Abschnitt Summary (Zusammenfassung) Edit (Bearbeiten).
5. Geben Sie eine neue Beschreibung im Dialogfeld ein und wählen Sie Save changes (Änderungen speichern).

Ändern der maximalen Sitzungsdauer einer Rolle (Konsole)

Um die Einstellung für die maximale Sitzungsdauer für Rollen anzugeben, die über die Konsole, die oder die AWS API übernommen werden AWS CLI, ändern Sie den Einstellungswert für die maximale Sitzungsdauer. Diese Einstellung kann einen Wert zwischen 1 Stunde und 12 Stunden haben. Wenn Sie keinen Wert angeben, wird der maximale Standardwert von einer Stunde angewendet. Diese Einstellung schränkt keine Sitzungen ein, die von AWS -Services übernommen werden.

Um die Einstellung für die maximale Sitzungsdauer für Rollen zu ändern AWS CLI, die über die Konsole oder AWS API (Konsole) übernommen werden

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 der IAM Console Roles (Rollen) aus.
3. Wählen Sie den Namen der zu ändernden Rolle.
4. Wählen Sie im Abschnitt Summary (Zusammenfassung), Edit (Bearbeiten).
5. Wählen Sie für Maximum session duration (Maximale Sitzungsdauer) einen Wert. Alternativ wählen Sie Custom duration (Benutzerdefinierte Dauer) und geben einen Wert (in Sekunden) ein.
6. Wählen Sie Änderungen speichern aus.

   Ihre Änderungen werden erst wirksam, wenn das nächste Mal jemand diese Rolle annimmt. Weitere Informationen dazu, wie Sie bestehende Sitzungen für diese Rolle widerrufen, finden Sie unter Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen.

In der AWS Management Console dauern IAM-Benutzersitzungen standardmäßig 12 Stunden. IAM-Benutzer, die in der Konsole die Rolle wechseln, erhalten die maximale Sitzungsdauer der Rolle oder die verbleibende Zeit in der Sitzung des Benutzers, je nachdem, was kürzer ist.

Jeder, der die Rolle über die AWS API AWS CLI oder übernimmt, kann bis zu diesem Maximum eine längere Sitzung beantragen. Die Einstellung `MaxSessionDuration` bestimmt die maximale Dauer der Rollensitzung, die angefordert werden kann.

- AWS CLI Verwenden Sie den `duration-seconds` Parameter, um eine Sitzungsdauer anzugeben. Weitere Informationen hierzu finden Sie unter Wechseln zu einer IAM-Rolle (AWS CLI).

- Verwenden Sie den `DurationSeconds` Parameter, um eine Sitzungsdauer mithilfe der AWS API anzugeben. Weitere Informationen hierzu finden Sie unter Zu einer IAM-Rolle (AWS API) wechseln.

Ändern einer Rollenberechtigungsgrenze (Konsole)

Um die maximalen Berechtigungen zu ändern, die für eine Rolle zulässig sind, ändern Sie die Berechtigungsgrenze der Rolle.

Ändern der Richtlinie zum Festlegen der Berechtigungsgrenze für eine Rolle

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 Rollen aus.
3. Wählen Sie den Namen der Rolle mit der Berechtigungsgrenze, die Sie ändern möchten.
4. Wählen Sie die Registerkarte Berechtigungen. Falls erforderlich, öffnen Sie den Abschnitt Permissions boundary (Berechtigungsgrenze) und wählen Change boundary (Grenze ändern).
5. Wählen Sie die Richtlinie aus, die Sie für die Berechtigungsgrenze verwenden möchten.
6. Wählen Sie Change boundary (Grenze ändern).

   Ihre Änderungen werden erst wirksam, wenn das nächste Mal jemand diese Rolle annimmt.

Ändern einer Rolle (AWS CLI)

Sie können den verwenden AWS Command Line Interface , um eine Rolle zu ändern. Informationen zum Ändern der Gruppe von Tags einer Rolle finden Sie unter Verwaltung von Tags in IAM-Rollen (AWS CLI oder AWS API).

Themen

- Ändern einer Rollenvertrauensrichtlinie (AWS CLI)
- Ändern einer Rollenberechtigungsrichtlinie (AWS CLI)
- Ändern einer Rollenbeschreibung (AWS CLI)
- Ändern der maximalen Sitzungsdauer einer Rolle (AWS CLI)
- Ändern einer Rollenberechtigungsgrenze (AWS CLI)

Ändern einer Rollenvertrauensrichtlinie (AWS CLI)

Um zu ändern, wer eine Rolle übernehmen kann, müssen Sie die Vertrauensrichtlinie der Rolle bearbeiten. Sie können die Vertrauensrichtlinie für eine serviceverknüpfte Rolle nicht ändern.

Hinweise

- Wenn ein Benutzer als Auftraggeber in der Vertrauensrichtlinie einer Rolle aufgeführt ist, aber die Rolle nicht übernehmen kann, überprüfen Sie die Berechtigungsgrenze des Benutzers. Wenn eine Berechtigungsgrenze für den Benutzer festgelegt ist, dann muss sie die `sts:AssumeRole`-Aktion zulassen.
- Damit Benutzer die aktuelle Rolle innerhalb einer Rollensitzung wieder übernehmen können, geben Sie den Rollen-ARN oder AWS-Konto ARN als Principal in der Rollenvertrauensrichtlinie an. AWS-Services die Rechenressourcen wie Amazon EC2, Amazon ECS, Amazon EKS und Lambda bereitstellen, stellen temporäre Anmeldeinformationen bereit und aktualisieren diese Anmeldeinformationen automatisch. Dadurch wird sichergestellt, dass Sie immer über gültige Anmeldeinformationen verfügen. Für diese Dienste ist es nicht erforderlich, die aktuelle Rolle erneut anzunehmen, um temporäre Anmeldeinformationen zu erhalten. Wenn Sie jedoch beabsichtigen, Sitzungs-Tags oder eine Sitzungsrichtlinie zu übergeben, müssen Sie die aktuelle Rolle erneut annehmen. Informationen zum Ändern einer Rollenvertrauensrichtlinie zum Hinzufügen der Hauptrollen ARN oder AWS-Konto ARN finden Sie unterÄndern einer Rollenvertrauensrichtlinie (Konsole).

So ändern Sie eine Rollenvertrauensrichtinie (AWS CLI)

1. (Optional) Wenn Sie den Namen der Rolle, die Sie ändern möchten, nicht kennen, führen Sie den folgenden Befehl aus, um die Rollen im Konto aufzulisten:
   - aws iam list-roles
2. (Optional) Um die aktuelle Vertrauensrichtlinie einer Rolle anzuzeigen, führen Sie den folgenden Befehl aus:
   - aws iam get-role

3. Um die vertrauenswürdigen Auftraggeber zu ändern, die auf die Rolle zugreifen können, erstellen Sie eine Textdatei mit der aktualisierten Vertrauensrichtlinie. Sie können zum Erstellen der Richtlinie einen beliebigen Texteditor verwenden.

   Die folgende Vertrauensrichtlinie zeigt beispielsweise, wie AWS-Konten in dem `Principal` Element auf zwei verwiesen wird. Auf diese Weise können Benutzer innerhalb von zwei separaten AWS-Konten diese Rolle übernehmen.

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Wenn Sie einen Auftraggeber in einem anderen Konto angeben, ist das Hinzufügen eines Kontos zur Vertrauensrichtlinie einer Rolle nur die Hälfte der Einrichtung einer kontoübergreifenden Vertrauensbeziehung. Standardmäßig kann kein Benutzer in den vertrauenswürdigen Konten die Rolle übernehmen. Der Administrator für das neue vertrauenswürdige Konto muss den Benutzern die Berechtigung erteilen, die Rolle zu übernehmen. Dazu muss der Administrator eine Richtlinie erstellen oder bearbeiten, die an den Benutzer angehängt ist, um dem Benutzer den Zugriff auf die Aktion `sts:AssumeRole` zu ermöglichen. Weitere Informationen finden Sie im folgenden Verfahren oder unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.
4. Um die soeben erstellte Datei zur Aktualisierung der Vertrauensrichtlinie zu verwenden, führen Sie den folgenden Befehl aus:

   - war ich update-assume-role-policy

So erteilen Sie Benutzern in einem vertrauenswürdigen Konto die Berechtigung zur Verwendung der Rolle (AWS CLI)

Weitere Informationen und Details zu dieser Vorgehensweise finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

1. Erstellen Sie eine JSON-Datei, die eine Berechtigungsrichtlinie enthält, die Berechtigungen für die Übernahme der Rolle erteilt. Die folgende Richtlinie enthält beispielsweise die erforderlichen Mindestberechtigungen:

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
     }
   }
   ```

   Ersetzen Sie den ARN in der Anweisung durch den ARN der Rolle, die der Benutzer übernehmen kann.

2. Führen Sie den folgenden Befehl aus, um die JSON-Datei, die die Vertrauensrichtlinie enthält, in IAM hochzuladen:

   - aws iam create-policy

   Die Ausgabe dieses Befehls enthält den ARN der Richtlinie. Notieren Sie sich diesen ARN, da Sie ihn zu einem späteren Zeitpunkt brauchen.

3. Legen Sie fest, welchem Benutzer oder welcher Gruppe Sie die Richtlinie anfügen. Wenn Sie den Namen des betreffenden Benutzers oder der Gruppe nicht kennen, geben Sie einen der folgenden Befehle ein, um die Benutzer oder Gruppen im Konto aufzulisten:

   - aws iam list-users
   - aws iam list-groups

4. Verwenden Sie einen der folgenden Befehle, um die im vorherigen Schritt erstellte Richtlinie an einen Benutzer oder eine Gruppe anzufügen:

   - war ich attach-user-policy
   - war ich attach-group-policy

Ändern einer Rollenberechtigungsrichtlinie (AWS CLI)

Um die von der Rolle zugelassenen Berechtigungen zu ändern, modifizieren Sie die Berechtigungsrichtlinie (oder Berechtigungsrichtlinien) der Rolle. Sie können die Berechtigungsrichtlinie für eine serviceverknüpfte Rolle in IAM nicht ändern. Möglicherweise können Sie die Berechtigungsrichtlinie innerhalb des Service ändern, der von der Rolle abhängt. Um zu überprüfen, ob ein Service dieses Feature unterstützt, lesen Sie AWS Dienste, die mit IAM funktionieren und suchen Sie nach den Services mit Yes (Ja) in der Spalte Service-linked roles (Serviceverknüpfte Rollen). Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

So ändern Sie die Berechtigungen einer Rolle (AWS CLI)

1. (Optional) Um die aktuell mit einer Rolle verknüpften Berechtigungen anzuzeigen, führen Sie die folgenden Befehle aus:

   1. aws ist bestrebt list-role-policies, Online-Richtlinien aufzulisten
   2. aws zielt darauf ab list-attached-role-policies, verwaltete Richtlinien aufzulisten
2. Der Befehl zum Aktualisieren von Berechtigungen der Rolle unterscheidet sich je nachdem, ob Sie eine verwaltete Inlinerichtlinie aktualisieren.

   Wenn Sie eine verwaltete Richtlinie aktualisieren möchten, führen Sie den folgenden Befehl aus, um eine neue Version der verwalteten Richtlinie zu erstellen:

   - war ich create-policy-version

   Um eine Inline-Richtlinie zu aktualisieren, führen Sie den folgenden Befehl aus:

   - war ich put-role-policy

Ändern einer Rollenbeschreibung (AWS CLI)

Um die Beschreibung einer Rolle zu ändern, modifizieren Sie den Text der Beschreibung.

So ändern Sie die Beschreibung einer Rolle (AWS CLI)

1. (Optional) Um die aktuelle Beschreibung einer Rolle anzuzeigen, führen Sie den folgenden Befehl aus:

- aws iam get-role

2. Um die Beschreibung einer Rolle zu aktualisieren, führen Sie den folgenden Befehl mit dem Beschreibungsparameter aus:

   - aws iam update-role

Ändern der maximalen Sitzungsdauer einer Rolle (AWS CLI)

Um die maximale Sitzungsdauer für Rollen festzulegen, die mithilfe der AWS CLI oder API angenommen werden, ändern Sie den Wert der maximalen Sitzungsdauer. Diese Einstellung kann einen Wert zwischen 1 Stunde und 12 Stunden haben. Wenn Sie keinen Wert angeben, wird der maximale Standardwert von einer Stunde angewendet. Diese Einstellung schränkt die von AWS Diensten angenommenen Sitzungen nicht ein.

> **Note**
>
> Jeder, der die Rolle von der API AWS CLI oder übernimmt, kann den `duration-seconds` CLI-Parameter oder den `DurationSeconds` API-Parameter verwenden, um eine längere Sitzung anzufordern. Die Einstellung `MaxSessionDuration` bestimmt die maximale Dauer der Rollensitzung, die mit dem `DurationSeconds`-Parameter angefordert werden kann. Wenn Benutzer keinen Wert für den `DurationSeconds`-Parameter angeben, sind ihre Sicherheitsanmeldeinformationen eine Stunde lang gültig.

So ändern Sie die maximale Sitzungsdauer für Rollen, für die unterstellt wird, dass sie die AWS CLI (AWS CLI) verwenden

1. (Optional) Um die aktuelle maximale Sitzungsdauer für eine Rolle anzuzeigen, führen Sie den folgenden Befehl aus:

   - aws iam get-role

2. Um die maximale Sitzungsdauer einer Rolle zu aktualisieren, führen Sie den folgenden Befehl mit dem CLI-Parameter `max-session-duration` oder dem API-Parameter `MaxSessionDuration` aus:

   - aws iam update-role

Ihre Änderungen werden erst wirksam, wenn das nächste Mal jemand diese Rolle annimmt. Weitere Informationen dazu, wie Sie bestehende Sitzungen für diese Rolle widerrufen, finden Sie unter Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen.

Ändern einer Rollenberechtigungsgrenze (AWS CLI)

Um die maximalen Berechtigungen zu ändern, die für eine Rolle zulässig sind, ändern Sie die Berechtigungsgrenze der Rolle.

Ändern der verwalteten Richtlinie zum Festlegen der Berechtigungsgrenze für eine Rolle (AWS CLI)

1. (Optional) Um die aktuelle Berechtigungsgrenze einer Rolle anzuzeigen, führen Sie den folgenden Befehl aus:

   - aws iam get-role

2. Um eine andere verwaltete Richtlinie zum Aktualisieren der Berechtigungsgrenze für eine Rolle zu verwenden, führen Sie den folgenden Befehl aus:

   - als ich put-role-permissions-boundary

   Eine Rolle kann nur eine verwaltete Richtlinie als Berechtigungsgrenze festlegen. Wenn Sie die Berechtigungsgrenze ändern, ändern Sie die maximal zulässigen Berechtigungen für eine Rolle.

Eine Rolle ändern (AWS API)

Sie können die AWS API verwenden, um eine Rolle zu ändern. Informationen zum Ändern der Gruppe von Tags einer Rolle finden Sie unter Verwaltung von Tags in IAM-Rollen (AWS CLI oder AWS API).

Themen

- Änderung einer Rollenvertrauensrichtlinie (AWS API)
- Änderung einer Rollenberechtigungsrichtlinie (AWS API)
- Ändern einer Rollenbeschreibung (AWS -API)
- Änderung der maximalen Sitzungsdauer einer Rolle (AWS API)
- Änderung einer Grenze für Rollenberechtigungen (AWS API)

Änderung einer Rollenvertrauensrichtlinie (AWS API)

Um zu ändern, wer eine Rolle übernehmen kann, müssen Sie die Vertrauensrichtlinie der Rolle bearbeiten. Sie können die Vertrauensrichtlinie für eine serviceverknüpfte Rolle nicht ändern.

> **Hinweise**
>
> - Wenn ein Benutzer als Auftraggeber in der Vertrauensrichtlinie einer Rolle aufgeführt ist, aber die Rolle nicht übernehmen kann, überprüfen Sie die Berechtigungsgrenze des Benutzers. Wenn eine Berechtigungsgrenze für den Benutzer festgelegt ist, dann muss sie die `sts:AssumeRole`-Aktion zulassen.
> - Damit Benutzer die aktuelle Rolle innerhalb einer Rollensitzung wieder übernehmen können, geben Sie den Rollen-ARN oder AWS-Konto ARN als Principal in der Rollenvertrauensrichtlinie an. AWS-Services die Rechenressourcen wie Amazon EC2, Amazon ECS, Amazon EKS und Lambda bereitstellen, stellen temporäre Anmeldeinformationen bereit und aktualisieren diese Anmeldeinformationen automatisch. Dadurch wird sichergestellt, dass Sie immer über gültige Anmeldeinformationen verfügen. Für diese Dienste ist es nicht erforderlich, die aktuelle Rolle erneut anzunehmen, um temporäre Anmeldeinformationen zu erhalten. Wenn Sie jedoch beabsichtigen, Sitzungs-Tags oder eine Sitzungsrichtlinie zu übergeben, müssen Sie die aktuelle Rolle erneut annehmen. Informationen zum Ändern einer Rollenvertrauensrichtlinie zum Hinzufügen der Hauptrollen ARN oder AWS-Konto ARN finden Sie unterÄndern einer Rollenvertrauensrichtlinie (Konsole).

So ändern Sie eine Rollenvertrauensrichtlinie (AWS API)

1. (Optional) Wenn Sie den Namen der Rolle, die Sie ändern möchten, nicht kennen, rufen Sie die folgende Operation auf, um die Rollen im Konto aufzulisten:
   - ListRoles
2. (Optional) Um die aktuelle Vertrauensrichtlinie einer Rolle anzuzeigen, rufen Sie die folgende Operation auf:
   - GetRole

3. Um die vertrauenswürdigen Auftraggeber zu ändern, die auf die Rolle zugreifen können, erstellen Sie eine Textdatei mit der aktualisierten Vertrauensrichtlinie. Sie können zum Erstellen der Richtlinie einen beliebigen Texteditor verwenden.

   Die folgende Vertrauensrichtlinie zeigt beispielsweise, wie AWS-Konten in dem `Principal` Element auf zwei verwiesen wird. Auf diese Weise können Benutzer innerhalb von zwei separaten AWS-Konten diese Rolle übernehmen.

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Wenn Sie einen Auftraggeber in einem anderen Konto angeben, ist das Hinzufügen eines Kontos zur Vertrauensrichtlinie einer Rolle nur die Hälfte der Einrichtung einer kontoübergreifenden Vertrauensbeziehung. Standardmäßig kann kein Benutzer in den vertrauenswürdigen Konten die Rolle übernehmen. Der Administrator für das neue vertrauenswürdige Konto muss den Benutzern die Berechtigung erteilen, die Rolle zu übernehmen. Dazu muss der Administrator eine Richtlinie erstellen oder bearbeiten, die an den Benutzer angehängt ist, um dem Benutzer den Zugriff auf die Aktion `sts:AssumeRole` zu ermöglichen. Weitere Informationen finden Sie im folgenden Verfahren oder unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.
4. Um die soeben erstellte Datei zur Aktualisierung der Vertrauensrichtlinie zu verwenden, führen Sie die folgende Operation aus:

   - UpdateAssumeRolePolicy

Um Benutzern in einem vertrauenswürdigen externen Konto die Verwendung der Rolle (AWS API) zu ermöglichen

Weitere Informationen und Details zu dieser Vorgehensweise finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

1. Erstellen Sie eine JSON-Datei, die eine Berechtigungsrichtlinie enthält, die Berechtigungen für die Übernahme der Rolle erteilt. Die folgende Richtlinie enthält beispielsweise die erforderlichen Mindestberechtigungen:

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
     }
   }
   ```

   Ersetzen Sie den ARN in der Anweisung durch den ARN der Rolle, die der Benutzer übernehmen kann.

2. Führen Sie die folgende Operation aus, um die JSON-Datei, die die Vertrauensrichtlinie enthält, in IAM hochzuladen:

   - CreatePolicy

   Die Ausgabe dieser Operation enthält den ARN der Richtlinie. Notieren Sie sich diesen ARN, da Sie ihn zu einem späteren Zeitpunkt brauchen.

3. Legen Sie fest, welchem Benutzer oder welcher Gruppe Sie die Richtlinie anfügen. Wenn Sie den Namen des betreffenden Benutzers oder der Gruppe nicht kennen, rufen Sie eine der folgenden Operationen auf, um die Benutzer oder Gruppen im Konto aufzulisten:

   - ListUsers
   - ListGroups

4. Rufen Sie eine der folgenden Operationen auf, um die im vorherigen Schritt erstellte Richtlinie an einen Benutzer oder eine Gruppe anzufügen:

   - API: AttachUserPolicy
   - AttachGroupPolicy

Änderung einer Rollenberechtigungsrichtlinie (AWS API)

Um die von der Rolle zugelassenen Berechtigungen zu ändern, modifizieren Sie die Berechtigungsrichtlinie (oder Berechtigungsrichtlinien) der Rolle. Sie können die Berechtigungsrichtlinie für eine serviceverknüpfte Rolle in IAM nicht ändern. Möglicherweise können Sie die Berechtigungsrichtlinie innerhalb des Service ändern, der von der Rolle abhängt. Um zu überprüfen, ob ein Service dieses Feature unterstützt, lesen Sie AWS Dienste, die mit IAM funktionieren und suchen Sie nach den Services mit Yes (Ja) in der Spalte Service-linked roles (Serviceverknüpfte Rollen). Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Um die von einer Rolle (AWS API) erlaubten Berechtigungen zu ändern

1. (Optional) Um die aktuell mit einer Rolle verknüpften Berechtigungen anzuzeigen, rufen Sie die folgenden Operationen auf:

   1. ListRolePoliciesum Inline-Richtlinien aufzulisten
   2. ListAttachedRolePoliciesum verwaltete Richtlinien aufzulisten
2. Die Operation zum Aktualisieren von Berechtigungen der Rolle unterscheidet sich je nachdem, ob Sie eine verwaltete oder eine Inline-Richtlinie aktualisieren.

   Wenn Sie eine verwaltete Richtlinie aktualisieren möchten, rufen Sie die folgende Operation auf, um eine neue Version der verwalteten Richtlinie zu erstellen:

   - CreatePolicyVersion

   Um eine Inline-Richtlinie zu aktualisieren, rufen Sie die folgende Operation auf:

   - PutRolePolicy

Ändern einer Rollenbeschreibung (AWS -API)

Um die Beschreibung einer Rolle zu ändern, modifizieren Sie den Text der Beschreibung.

Um die Beschreibung einer Rolle (AWS API) zu ändern

1. (Optional) Um die aktuelle Beschreibung einer Rolle anzuzeigen, rufen Sie die folgende Operation auf:

- GetRole

2. Um die Beschreibung einer Rolle zu aktualisieren, rufen Sie die folgende Operation mit dem Beschreibungsparameter auf:

   - UpdateRole

Änderung der maximalen Sitzungsdauer einer Rolle (AWS API)

Um die maximale Sitzungsdauer für Rollen festzulegen, die mithilfe der AWS CLI oder API angenommen werden, ändern Sie den Wert der maximalen Sitzungsdauer. Diese Einstellung kann einen Wert zwischen 1 Stunde und 12 Stunden haben. Wenn Sie keinen Wert angeben, wird der maximale Standardwert von einer Stunde angewendet. Diese Einstellung schränkt die von AWS Diensten angenommenen Sitzungen nicht ein.

> **Note**
>
> Jeder, der die Rolle von der API AWS CLI oder übernimmt, kann den `duration-seconds` CLI-Parameter oder den `DurationSeconds` API-Parameter verwenden, um eine längere Sitzung anzufordern. Die Einstellung `MaxSessionDuration` bestimmt die maximale Dauer der Rollensitzung, die mit dem `DurationSeconds`-Parameter angefordert werden kann. Wenn Benutzer keinen Wert für den `DurationSeconds`-Parameter angeben, sind ihre Sicherheitsanmeldeinformationen eine Stunde lang gültig.

Um die Einstellung für die maximale Sitzungsdauer für Rollen zu ändern, die mithilfe der API (AWS API) übernommen werden

1. (Optional) Um die aktuelle maximale Sitzungsdauer für eine Rolle anzuzeigen, rufen Sie die folgende Operation auf:

   - GetRole

2. Um die maximale Sitzungsdauer einer Rolle zu aktualisieren, rufen Sie die folgende Operation mit dem CLI-Parameter `max-sessionduration` oder dem API-Parameter `MaxSessionDuration` auf:

   - UpdateRole

Ihre Änderungen werden erst wirksam, wenn das nächste Mal jemand diese Rolle annimmt. Weitere Informationen dazu, wie Sie bestehende Sitzungen für diese Rolle widerrufen, finden Sie unter Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen.

Änderung einer Grenze für Rollenberechtigungen (AWS API)

Um die maximalen Berechtigungen zu ändern, die für eine Rolle zulässig sind, ändern Sie die Berechtigungsgrenze der Rolle.

Ändern der verwalteten Richtlinie zum Festlegen der Berechtigungsgrenze für eine Rolle (AWS -API)

1. (Optional) Um die aktuelle Berechtigungsgrenze einer Rolle anzuzeigen, führen Sie die folgende Operation aus:
   - GetRole
2. Um eine andere verwaltete Richtlinie zum Aktualisieren der Berechtigungsgrenze für eine Rolle zu verwenden, führen Sie die folgende Operation aus:
   - PutRolePermissionsBoundary

   Eine Rolle kann nur eine verwaltete Richtlinie als Berechtigungsgrenze festlegen. Wenn Sie die Berechtigungsgrenze ändern, ändern Sie die maximal zulässigen Berechtigungen für eine Rolle.

Löschen von Rollen oder Instance-Profilen

Wenn Sie eine Rolle nicht mehr benötigen, empfehlen wir, dass Sie die Rolle und die zugehörigen Berechtigungen löschen. Auf diese Weise haben Sie keine ungenutzte Entität, die nicht aktiv überwacht oder verwaltet wird.

Wenn die Rolle einer EC2-Instance zugeordnet wurde, können Sie die Rolle auch aus dem Instance-Profil entfernen und dieses dann löschen.

> **Warning**
>
> Stellen Sie sicher, dass keine Amazon EC2 Instances mit der Rolle oder dem Instance-Profil ausgeführt werden, die Sie löschen möchten. Wenn Sie eine Rolle oder ein Instanceprofil

löschen, das einer laufenden Instance zugeordnet ist, werden alle Anwendungen unterbrochen, die auf der Instance ausgeführt werden.

Wenn Sie eine Rolle nicht dauerhaft löschen möchten, können Sie sie deaktivieren. Ändern Sie dazu die Richtlinien der Rolle und widerrufen Sie dann alle aktuellen Sitzungen. Sie könnten der Rolle beispielsweise eine Richtlinie hinzufügen, die allen den Zugriff verweigert. AWS Sie können die Vertrauensrichtlinie auch bearbeiten, um jeder Person, die versucht, die Rolle zu übernehmen, den Zugriff zu verweigern. Weitere Informationen zum Widerrufen von Sitzungen finden Sie unter Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen.

Themen

Anzeigen des Rollenzugriffs

Bevor Sie eine Rolle löschen, sollten Sie überprüfen, wann die Rolle zuletzt verwendet wurde. Sie können dazu die AWS Management Console AWS CLI, oder die AWS API verwenden. Sie sollten sich diese Informationen ansehen, weil Sie nicht jemandem den Zugriff entziehen wollen, der gerade die Rolle verwendet.

Das Datum der letzten Aktivität der Rolle stimmt möglicherweise nicht mit dem letzten auf der Registerkarte Access Advisor gemeldeten Datum überein. Auf der Registerkarte Access Advisor werden Aktivitäten nur für Services gemeldet, die laut der Berechtigungsrichtlinien der Rolle zulässig sind. Das Datum der letzten Aktivität der Rolle beinhaltet den letzten Versuch, auf einen Dienst in zuzugreifen AWS.

> **Note**
>
> Der Nachverfolgungs-Zeitraum für die letzte Aktivität einer Rolle und die Access-Advisor-Daten umfassen die letzten 400 Tage. Dieser Zeitraum kann kürzer sein, wenn Ihre Region

diese Funktionen innerhalb des letzten Jahres zu unterstützen begonnen hat. Die Rolle könnte dann vor mehr als 400 Tagen verwendet worden sein. Weitere Informationen zum Nachverfolgungszeitraum finden Sie unter Wo werden die AWS zuletzt aufgerufenen Informationen aufgezeichnet.

So zeigen Sie an, wann eine Rolle zuletzt verwendet wurde (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. Wählen Sie im Navigationsbereich Rollen aus.
3. Suchen Sie die Zeile der Rolle, deren Aktivität Sie anzeigen möchten. Sie können das Suchfeld verwenden, um die Ergebnisse einzuschränken. Zeigen Sie die Spalte Last activity (Letzte Aktivität) an, um die Anzahl der Tage seit der letzten Verwendung der Rolle anzuzeigen. Wenn die Rolle nicht innerhalb des Verfolgungszeitraums verwendet wurde, wird in der Tabelle None (Keine)angezeigt.
4. Wählen Sie den Namen der Rolle, um weitere Informationen anzuzeigen. Die Seite Summary (Zusammenfassung) der Rolle enthält auch Last activity (Letzte Aktivität), mit der Anzeige des Datums, an dem die Rolle zuletzt verwendet wurde. Wenn die Rolle innerhalb der letzten 400 Tage nicht verwendet wurde, zeigt Last activity (Letzte Aktivität) Not accessed in the tracking period (Kein Zugriff im nachverfolgungszeitraum) an.

So zeigen Sie an, wann eine Rolle zuletzt verwendet wurde (AWS CLI):

`aws iam get-role` - Führen Sie diesen Befehl aus, um Informationen über eine Rolle, einschließlich des `RoleLastUsed`-Objekts, zurückzugeben. Dieses Objekt enthält die `LastUsedDate` und die `Region`, in denen die Rolle zuletzt verwendet wurde. Wenn `RoleLastUsed` vorhanden ist, aber keinen Wert enthält, wurde die Rolle innerhalb des Verfolgungszeitraums nicht verwendet.

Um zu sehen, wann eine Rolle zuletzt verwendet wurde (AWS API)

`GetRole` - Rufen Sie diesen Vorgang auf, um Informationen über eine Rolle, einschließlich des `RoleLastUsed`-Objekts, zurückzugeben. Dieses Objekt enthält die `LastUsedDate` und die `Region`, in denen die Rolle zuletzt verwendet wurde. Wenn `RoleLastUsed` vorhanden ist, aber keinen Wert enthält, wurde die Rolle innerhalb des Verfolgungszeitraums nicht verwendet.

Löschen einer serviceverknüpften Rolle

Wenn die Rolle eine serviceverknüpfte Rolle ist, überprüfen Sie die Dokumentation für den verknüpften Service, um zu erfahren, wie Sie die Rolle löschen können. Sie können die serviceverknüpften Rollen in Ihrem Konto anzeigen, indem Sie zur IAM-Seite Roles der Konsole wechseln. Serviceverknüpfte Rollen werden mit dem Hinweis (Service-linked role) in der Spalte Trusted entities (Vertrauenswürdige Entitäten) der Tabelle angezeigt. Ein Banner auf der Seite Summary (Übersicht) für die Rolle zeigt ebenfalls an, dass es sich um eine serviceverknüpfte Rolle handelt.

Wenn der Service keine Dokumentation zum Löschen der mit dem Service verknüpften Rolle enthält, können Sie die Rolle mithilfe der IAM-Konsole oder der API löschen. AWS CLI Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle.

Löschen einer IAM-Rolle (Konsole)

Wenn Sie die AWS Management Console zum Löschen einer Rolle verwenden, trennt IAM automatisch die mit der Rolle verknüpften verwalteten Richtlinien. Außerdem werden alle eingebundenen Richtlinien gelöscht, die mit der Rolle verbunden sind. Außerdem werden alle Amazon-EC2-Instance-Profile gelöscht, die die Rolle enthalten.

> **Important**
>
> In einigen Fällen kann eine Rolle mit einem Amazon EC2-Instance-Profil verbunden sein, und die Rolle und das Instance-Profil können denselben Namen haben. In diesem Fall können Sie die AWS Management Console zum Löschen der Rolle und des Instanzprofils verwenden. Diese Verknüpfung erfolgt automatisch für Rollen und Instance-Profile, die Sie in der Konsole erstellen. Wenn Sie die Rolle über Tools für Windows PowerShell oder die AWS API erstellt haben, haben die Rolle und das Instanzprofil möglicherweise unterschiedliche Namen. AWS CLI In diesem Fall können Sie die Konsole nicht zum Löschen verwenden. Stattdessen müssen Sie Tools für Windows oder AWS API verwenden PowerShell, um die Rolle zunächst aus dem Instanzprofil zu entfernen. AWS CLI Sie müssen die Rolle dann in einem separaten Schritt löschen.

So löschen Sie eine Rolle (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. Wählen Sie im Navigationsbereich Roles (Rollen), und aktivieren Sie dann das Kontrollkästchen neben der Rolle, die Sie löschen möchten.
3. Wählen Sie oben auf der Seite Delete role (Rolle löschen).
4. Überprüfen Sie im Bestätigungsdialogfeld die Informationen, auf die zuletzt zugegriffen wurde. Darin wird angegeben, wann jede der ausgewählten Rollen zuletzt auf einen AWS Dienst zugegriffen hat. Auf diese Weise können Sie leichter bestätigen, ob die Rolle derzeit aktiv ist. Wenn Sie fortfahren möchten, geben Sie den Namen der Rolle in das Texteingabefeld ein und wählen Sie Löschen. Wenn Sie sich sicher sind, können Sie mit dem Löschen fortfahren, auch wenn die Informationen zum letzten Zugriff noch geladen werden.

> **Note**
>
> Sie können die Konsole nicht verwenden, um ein Instance-Profil zu löschen, es sei denn, es hat denselben Namen wie die Rolle. Außerdem müssen Sie das Instance-Profil im Rahmen des Vorgangs zum Löschen einer Rolle, wie im zuvor genannten Vorgang beschrieben, löschen. Um ein Instanzprofil zu löschen, ohne auch die Rolle zu löschen, müssen Sie die AWS API AWS CLI oder verwenden. Weitere Informationen finden Sie in den folgenden Abschnitten.

Erstellen einer IAM-Rolle (AWS CLI)

Wenn Sie die verwenden, AWS CLI um eine Rolle zu löschen, müssen Sie zuerst die mit der Rolle verknüpften Inline-Richtlinien löschen. Sie müssen auch die mit der Rolle verknüpften verwalteten Richtlinien trennen. Wenn Sie das verbundene Instance-Profil, das die Rolle enthält, löschen möchten, müssen Sie es separat löschen.

So löschen Sie eine Rolle (AWS CLI)

1. Wenn Sie den Namen der Rolle, die Sie löschen möchten, nicht kennen, geben Sie den folgenden Befehl ein, um die Rollen in Ihrem Konto aufzulisten:

   ```
   aws iam list-roles
   ```

   Die Liste enthält den Amazon-Ressourcennamen (ARN) jeder Rolle. Verwenden Sie den Rollennamen, nicht den ARN, um sich auf Rollen mit den CLI-Befehlen zu beziehen. Wenn

eine Rolle zum Beispiel folgenden ARN hat: `arn:aws:iam::123456789012:role/myrole`, verweisen Sie auf die Rolle als **`myrole`**.

2. Entfernen Sie die Rolle aus allen Instance-Profilen, mit denen die Rolle verbunden ist.

   a. Um alle Instance-Profile aufzulisten, denen die Rolle zugeordnet ist, geben Sie den folgenden Befehl ein:

   ```
   aws iam list-instance-profiles-for-role --role-name role-name
   ```

   b. Um die Rolle aus einem Instance-Profil zu entfernen, geben Sie den folgenden Befehl für jedes Instance-Profil ein:

   ```
   aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
   ```

3. Löschen Sie alle Richtlinien, die mit der Rolle verbunden sind.

   a. Um alle Inline-Richtlinien aufzulisten, die in der Rolle vorhanden sind, geben Sie den folgenden Befehl ein:

   ```
   aws iam list-role-policies --role-name role-name
   ```

   b. Um jede Inline-Richtlinie aus der Rolle zu löschen, geben Sie den folgenden Befehl für jede Richtlinie ein:

   ```
   aws iam delete-role-policy --role-name role-name --policy-name policy-name
   ```

   c. Um alle verwalteten Richtlinien aufzulisten, die der Rolle angefügt sind, geben Sie den folgenden Befehl ein:

   ```
   aws iam list-attached-role-policies --role-name role-name
   ```

   d. Um jede verwaltete Richtlinie von der Rolle zu trennen, geben Sie den folgenden Befehl für jede Richtlinie ein:

   ```
   aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
   ```

4. Geben Sie den folgenden Befehl ein, um die Rolle zu löschen:

   ```
   aws iam delete-role --role-name role-name
   ```

5. Wenn Sie nicht vorhaben, die Instance-Profile, die der Rolle zugeordnet waren, wiederzuverwenden, geben Sie den folgenden Befehl ein, um sie zu löschen:

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Löschen einer IAM-Rolle (AWS -API)

Wenn Sie über die IAM-API eine Rolle löschen, müssen Sie zuerst die Inline-Richtlinien löschen, die der Rolle zugeordnet sind. Sie müssen auch die mit der Rolle verknüpften verwalteten Richtlinien trennen. Wenn Sie das verbundene Instance-Profil, das die Rolle enthält, löschen möchten, müssen Sie es separat löschen.

Um eine Rolle zu löschen (AWS API)

1. Rufen Sie auf, um alle Instanzprofile aufzulisten, denen eine Rolle zugeordnet ist ListInstanceProfilesForRole.

   Rufen Sie an, um die Rolle aus einem Instanzprofil zu entfernen RemoveRoleFromInstanceProfile. Sie müssen den Rollennamen und den Instance-Profilnamen übergeben.

   Wenn Sie ein Instanzprofil, das mit der Rolle verknüpft war, nicht wiederverwenden möchten, rufen Sie an, DeleteInstanceProfileum es zu löschen.
2. Rufen Sie an, um alle Inline-Richtlinien für eine Rolle aufzulisten ListRolePolicies.

   Rufen Sie an, um Inline-Richtlinien zu löschen, die der Rolle zugeordnet sind DeleteRolePolicy. Sie müssen den Rollennamen und den Inline-Richtliniennamen übergeben.
3. Rufen Sie an, um alle verwalteten Richtlinien aufzulisten, die einer Rolle zugeordnet sind ListAttachedRolePolicies.

   Rufen DetachRolePolicySie an, um verwaltete Richtlinien, die der Rolle zugeordnet sind, zu trennen. Sie müssen den Rollennamen und den ARN der verwalteten Richtlinie übergeben.
4. Rufen Sie an DeleteRole, um die Rolle zu löschen.

Ähnliche Informationen

Allgemeine Informationen zu Instance-Profilen finden Sie unter Verwenden von Instance-Profilen.

Allgemeine Informationen zu serviceverknüpften Rollen finden Sie unter Verwenden von serviceverknüpften Rollen.

Identitätsanbieter und Verbund

Wenn Sie bereits Benutzeridentitäten außerhalb von verwalten AWS, können Sie Identitätsanbieter verwenden, anstatt IAM-Benutzer in Ihrem zu erstellen. AWS-Konto Mit einem Identitätsanbieter (IdP) können Sie Ihre Benutzeridentitäten außerhalb von verwalten AWS und diesen externen Benutzeridentitäten Berechtigungen zur Nutzung von AWS Ressourcen in Ihrem Konto erteilen. Dies ist hilfreich, wenn Sie in Ihrer Organisation bereits ein eigenes Identitätssystem wie ein unternehmensweites Benutzerverzeichnis verwenden. Außerdem kann es beim Erstellen einer mobilen App oder Webanwendung nützlich sein, die Zugriff auf AWS -Ressourcen benötigt.

Ein externer IdP stellt Identitätsinformationen für die AWS Verwendung von OpenID Connect (OIDC) oder SAML 2.0 (Security Assertion Markup Language 2.0) bereit. OIDC verbindet Anwendungen wie GitHub Aktionen, die nicht auf Ressourcen ausgeführt werden. AWS AWS Beispiele für bekannte SAML-Identitätsanbieter sind Shibboleth und Active Directory Federation Services.

> **Note**
>
> Als bewährte Sicherheitsmethode empfehlen wir, menschliche Benutzer in IAM Identity Center mit einem externen SAML-Identitätsanbieter zu verwalten, anstatt einen SAML-Verbund in IAM zu erstellen. Informationen zu bestimmten Situationen, in denen ein IAM-Benutzer erforderlich ist, finden Sie unter Wann sollte ein IAM-Benutzer (anstelle einer Rolle) erstellt werden?.

Wenn Sie einen -Identitätsanbieter verwenden, müssen Sie keinen eigenen Anmeldecode schreiben oder eigene Benutzeridentitäten verwalten. Der Identitätsanbieter erledigt das für Sie. Ihre externen Benutzer melden sich über einen IdP an, und Sie können diesen externen Identitäten Berechtigungen zur Nutzung von AWS Ressourcen in Ihrem Konto erteilen. Identitätsanbieter helfen Ihnen dabei, Ihre AWS-Konto Sicherheit zu gewährleisten, da Sie keine langfristigen Sicherheitsanmeldedaten wie Zugriffsschlüssel verteilen oder in Ihre Anwendung einbetten müssen.

In diesem Handbuch wird der IAM-Verbund behandelt. Ihr Anwendungsfall wird möglicherweise besser von IAM Identity Center oder Amazon Cognito unterstützt. Die folgenden Zusammenfassungen und Tabellen bieten einen Überblick über die Methoden, mit denen Ihre Benutzer Verbundzugriff auf AWS Ressourcen erhalten können.

	Account type (Art des Kontos)	Zugriffsverwaltung von:	Unterstützte Identität squelle
Verbund mit IAM Identity Center	Mehrere Konten, verwaltet von AWS Organizations	Die menschlichen Benutzer Ihrer Belegschaft	• SAML 2.0 • Verwaltetes Active Directory • Identity-Center-Verzeichnis
Verbund mit IAM	Einzelnes, eigenständiges Konto	• Menschliche Benutzer in kurzfristigen, kleinen Bereitstellungen • Maschinelle Benutzer	• SAML 2.0 • OIDC
Verwenden von Amazon Cognito Sync mit Identitätspools	Any	Die Benutzer von Apps, die für den Zugriff auf Ressourcen eine IAM-Autorisierung benötigen	• SAML 2.0 • OIDC • Bestimmte OAuth-2.0-Social-Identity-Anbieter

Verbund mit IAM Identity Center

Für eine zentralisierte Zugriffsverwaltung von menschlichen Benutzern empfehlen wir Ihnen die Verwendung von IAM Identity Center, um den Zugriff auf Ihre Konten und die Berechtigungen innerhalb dieser Konten zu verwalten. Benutzern im IAM Identity Center werden kurzfristig Zugangsdaten für Ihre AWS Ressourcen gewährt. Sie können Active Directory, einen externen Identitätsanbieter (IdP) oder ein IAM Identity Center-Verzeichnis als Identitätsquelle für Benutzer und Gruppen verwenden, um Zugriff auf Ihre AWS Ressourcen zuzuweisen.

IAM Identity Center unterstützt den Identitätsverbund mit SAML (Security Assertion Markup Language) 2.0, um Benutzern, die berechtigt sind, Anwendungen innerhalb des Zugriffsportals zu verwenden, einen föderierten Single Sign-On-Zugriff zu bieten. AWS Benutzer können sich dann per Single Sign-On bei Diensten anmelden, die SAML unterstützen, einschließlich Anwendungen

AWS Management Console und Drittanbieteranwendungen wie Microsoft 365, SAP Concur und Salesforce.

Verbund mit IAM

Wir empfehlen zwar dringend, menschliche Benutzer in IAM Identity Center zu verwalten, aber Sie können den Verbundbenutzerzugriff mit IAM für menschliche Benutzer in kurzfristigen, kleinen Bereitstellungen aktivieren. Mit IAM können Sie separate SAML 2.0 und Open ID Connect (OIDC) verwenden IdPs und föderierte Benutzerattribute für die Zugriffskontrolle verwenden. Mit IAM können Sie Benutzerattribute wie Kostenstelle, Titel oder Gebietsschema von Ihnen an übergeben und detaillierte Zugriffsberechtigungen IdPs auf AWS der Grundlage dieser Attribute implementieren.

Eine Workload ist eine Sammlung von Ressourcen und Code, die einen geschäftlichen Nutzen erbringen, z. B. eine Anwendung oder ein Backend-Prozess. Ihr Workload kann eine IAM-Identität erfordern, um Anfragen an AWS Dienste, Anwendungen, Betriebstools und Komponenten zu stellen. Zu diesen Identitäten gehören Maschinen, die in Ihren AWS Umgebungen ausgeführt werden, z. B. Amazon EC2 EC2-Instances oder AWS Lambda -Funktionen.

Sie können auch Maschinenidentitäten für externe Parteien verwalten, die Zugriff benötigen. Um Zugriff auf Maschinenidentitäten zu gewähren, können Sie IAM-Rollen verwenden. IAM-Rollen verfügen über spezifische Berechtigungen und ermöglichen den Zugriff, AWS indem sie sich bei einer Rollensitzung auf temporäre Sicherheitsanmeldedaten verlassen. Darüber hinaus benötigen möglicherweise Computer außerhalb AWS dieser Systeme Zugriff auf Ihre AWS Umgebungen. Für Maschinen, die außerhalb von AWS Ihnen laufen, können Sie IAM Roles Anywhere verwenden. Weitere Informationen zu Rollen finden Sie unter IAM-Rollen. Einzelheiten dazu, wie Sie Rollen verwenden können, um den Zugriff an andere zu delegieren AWS-Konten, finden Sie unter. Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen

Um einen IdP direkt mit IAM zu verknüpfen, erstellen Sie eine Identitätsanbieter-Entität, um eine Vertrauensbeziehung zwischen Ihnen AWS-Konto und dem IdP aufzubauen. IAM-Unterstützungen IdPs , die mit OpenID Connect (OIDC) oder SAML 2.0 (Security Assertion Markup Language 2.0) kompatibel sind. Weitere Informationen zur Verwendung einer dieser Optionen mit finden Sie in den folgenden IdPs Abschnitten: AWS

- OIDC-Föderation
- SAML 2.0-Verbund

Verwenden von Amazon Cognito Sync mit Identitätspools

Amazon Cognito wurde für Entwickler entwickelt, die Benutzer in ihren Mobil- und Web-Apps authentifizieren und autorisieren möchten. Amazon-Cognito-Benutzerpools fügen Ihrer App Anmelde- und Registrierungsfunktionen hinzu, und Identitätspools stellen IAM-Anmeldeinformationen bereit, die Ihren Benutzern Zugriff auf geschützte Ressourcen gewähren, die Sie in AWS verwalten. Identitätspools erwerben über den API-Vorgang `AssumeRoleWithWebIdentity` Anmeldeinformationen für temporäre Sitzungen.

Amazon Cognito arbeitet mit externen Identitätsanbietern zusammen, die SAML und OpenID Connect unterstützen, sowie mit Anbietern sozialer Identitäten wie Facebook, Google und Amazon. Ihre App kann einen Benutzer mit einem Benutzerpool oder einem externen IdP anmelden und dann Ressourcen in seinem Namen mit benutzerdefinierten temporären Sitzungen in einer IAM-Rolle abrufen.

Gängige Szenarien

> **Note**
>
> Wir empfehlen, dass Sie von Ihren menschlichen Benutzern verlangen, dass sie beim Zugriff AWS temporäre Anmeldeinformationen verwenden. Haben Sie darüber nachgedacht, es zu verwenden AWS IAM Identity Center? Sie können IAM Identity Center verwenden, um den Zugriff auf mehrere Konten zentral zu verwalten AWS-Konten und Benutzern MFA-geschützten Single Sign-On-Zugriff auf alle ihnen zugewiesenen Konten von einem Ort aus zu gewähren. Mit IAM Identity Center können Sie Benutzeridentitäten in IAM Identity Center erstellen und verwalten oder einfach eine Verbindung zu Ihrem vorhandenen SAML-2.0-kompatiblen Identitätsanbieter herstellen. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

Sie können einen externen Identitätsanbieter (IdP) verwenden, um Benutzeridentitäten außerhalb AWS von. und des externen IdP zu verwalten. Ein externer IdP kann Identitätsinformationen entweder AWS mithilfe von OpenID Connect (OIDC) oder Security Assertion Markup Language (SAML) bereitstellen. OIDC wird häufig verwendet, wenn eine Anwendung, die nicht ausgeführt wird, Zugriff auf Ressourcen benötigt. AWS AWS

Wenn Sie den Verbund mit einem externen IdP konfigurieren möchten, erstellen Sie einen IAM-Identitätsanbieter, um AWS über den externen IdP und seine Konfiguration zu informieren. Dies

schafft Vertrauen zwischen Ihrem AWS-Konto und dem externen IdP. Die folgenden Themen enthalten allgemeine Szenarien für die Verwendung von IAM-Identitätsanbietern.

Themen

- Verwenden von Amazon Cognito für mobile Apps
- Verwenden von OIDC-Verbund-API-Vorgängen für mobile Apps

Verwenden von Amazon Cognito für mobile Apps

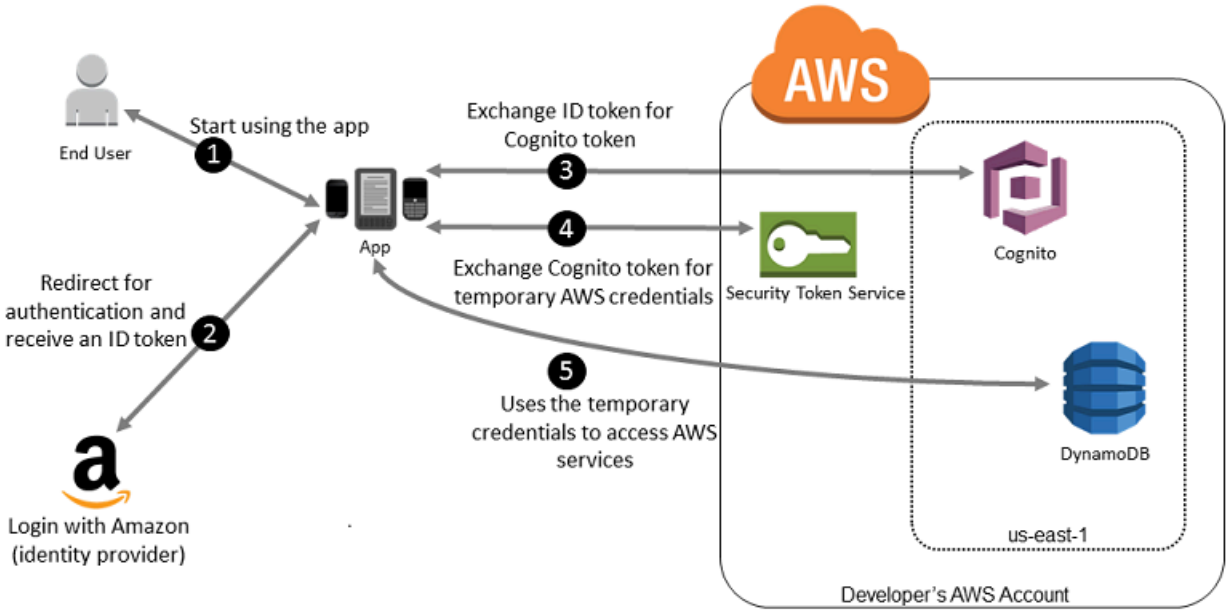
Die bevorzugte Methode zur Verwendung des OIDC-Verbunds ist die Verwendung von Amazon Cognito. Beispiel: Die Entwicklerin Adele entwickelt ein Spiel für ein Mobilgerät, in dem Benutzerdaten wie Spielstände und Profile in Amazon S3 und Amazon DynamoDB. gespeichert werden. Adele könnte diese Daten lokal auf dem Gerät speichern und sie mithilfe von Amazon Cognito zwischen Geräten synchronisieren. Sie weiß, dass es aus Sicherheits- und Wartungsgründen nicht sinnvoll ist, langfristige AWS -Sicherheitsanmeldeinformationen mit dem Spiel herauszugeben. Sie weiß auch, dass das Spiel möglicherweise eine große Anzahl von Benutzern haben wird. Aus all diesen Gründen möchte Sie in IAM nicht für jeden einzelnen Spieler eine neue Benutzeridentität anlegen. Stattdessen konzipiert sie das Spiel so, dass Benutzer sich mit einer bereits vorhandenen Identität eines bekannten externen Identitätsanbieters (Identity Provider, IdP) wie Login with Amazon, Facebook, Google oder eines anderen OpenID Connect (OIDC)-kompatiblen Identitätsanbieters anmelden können. In ihrem Spiel nutzt sie den Vorteil des Authentifizierungsmechanismus eines dieser Anbieter, um die Benutzeridentität zu überprüfen.

Damit die mobile App auf ihre AWS Ressourcen zugreifen kann, registriert sich Adele zunächst für eine Entwickler-ID bei der von ihr ausgewählten Person. IdPs Außerdem konfiguriert sie die Anwendung bei jedem dieser Anbieter. In ihrem Buch AWS-Konto , das den Amazon S3 S3-Bucket und die DynamoDB-Tabelle für das Spiel enthält, verwendet Adele Amazon Cognito, um IAM-Rollen zu erstellen, die genau die Berechtigungen definieren, die das Spiel benötigt. Wenn sie einen OIDC-IdP verwendet, erstellt sie auch eine IAM-OIDC-Identitätsanbieter-Entität, um Vertrauen zwischen einem Amazon Cognito Cognito-Identitätspool in ihr und dem IdP herzustellen. AWS-Konto

Im Code der App ruft Adele die Anmeldeschnittstelle des zuvor konfigurierten Identitätsanbieters auf. Der Identitätsanbieter übernimmt die Details der Anmeldung und die App erhält ein OAuth-Zugriffs-Token oder ein OIDC-ID-Token vom Anbieter. Adeles App kann diese Authentifizierungsinformationen gegen eine Reihe temporärer Sicherheitsanmeldedaten eintauschen, die aus einer AWS Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sitzungstoken bestehen. Die App kann diese Anmeldeinformationen dann verwenden, um auf Webdienste

zuzugreifen, die von AWS angeboten werden. Dabei ist die App auf die Berechtigungen beschränkt, die in der angenommenen Rolle festgelegt wurden.

Die nachfolgende Abbildung zeigt eine vereinfachte Darstellung dieses Vorgangs mit Anmeldung über Login with Amazon als Identitätsanbieter. In Schritt 2 kann die App auch Facebook, Google oder einen anderen OIDC-kompatiblen Identitätsanwender nutzen, was in der Abbildung jedoch nicht dargestellt ist.



1. Ein Kunde startet Ihre App auf einem Mobilgerät. Er wird aufgefordert, sich anzumelden.
2. Die App verwendet die Anmeldung über Login with Amazon, um die Anmeldeinformationen des Benutzers zu überprüfen.
3. Die App verwendet die Amazon-Cognito-API-Operationen `GetId` und `GetCredentialsForIdentity` um das Login-with-Amazon-ID-Token gegen ein Amazon-Cognito-Token auszutauschen. Amazon Cognito, das so konfiguriert wurde, dass es Ihrem Login-with-Amazon-Projekt vertraut, generiert ein Token, das es für temporäre Sitzungs-Anmeldeinformationen mit AWS STS austauscht.
4. Die App erhält temporäre Sicherheitsanmeldeinformationen von Amazon Cognito. Ihre App kann auch den Basic (Classic) -Workflow in Amazon Cognito verwenden, um Tokens von AWS STS Using abzurufen. `AssumeRoleWithWebIdentity` Weitere Informationen finden Sie unter Identitätspools (Verbundidentitäten) Authentifizierungsablauf im Amazon-Cognito-Entwicklerhandbuch.

5. Über diese temporären Sicherheitsanmeldeinformationen kann die App auf alle AWS -Ressourcen zugreifen, die für die App benötigt werden. Über die mit den temporären Sicherheitsanmeldeinformationen verknüpfte Rolle sowie die ihr zugewiesenen Richtlinien wird genau festgelegt, welcher Zugriff gewährt wird.

Gehen Sie wie folgt vor, um Ihre App so zu konfigurieren, dass sie Amazon Cognito verwendet, um Benutzer zu authentifizieren und Ihrer App Zugriff auf Ressourcen zu AWS gewähren. Wie Sie dieses Szenario genau umsetzen, wird in der Dokumentation zu Amazon Cognito erläutert.

1. (Optional) Melden Sie sich als Entwickler bei Login with Amazon, Facebook, Google oder einem anderen OpenID Connect (OIDC)-kompatiblen Identitätsanbieter an und konfigurieren Sie eine oder mehrere Anwendungen mit dem Anbieter. Dieser Schritt ist optional, da Amazon Cognito auch unauthentifizierten (Gast-)Zugang für Ihre Benutzer unterstützt.
2. Gehen Sie zu Amazon Cognito in der AWS Management Console. Verwenden Sie den Amazon Cognito-Assistenten, um einen Identitätspool zu erstellen. Dabei handelt es sich um einen Container, den Amazon Cognito verwendet, um die Identitäten der Endbenutzer für Ihre Anwendungen zu organisieren. Identitäten-Pools können zwischen Apps geteilt werden. Wenn Sie einen Identitätspool einrichten, erstellt Amazon Cognito eine oder zwei IAM-Rollen (eine für authentifizierte Identitäten und eine für nicht authentifizierte "Gast"-Identitäten), die Berechtigungen für Amazon Cognito-Benutzer definieren.
3. Integrieren Sie AWSAmplify in Ihre App und importieren Sie die Dateien, die für die Verwendung von Amazon Cognito erforderlich sind.
4. Erstellen Sie eine Instance des Amazon Cognito Credentials Providers und übergeben Sie die Identitätspool-ID, Ihre AWS-Konto -Nummer und den Amazon-Ressourcenname (ARN) der Rollen, die Sie mit dem Identitätspool verknüpft haben. Der Amazon Cognito-Assistent im AWS Management Console bietet Beispielcode, der Ihnen den Einstieg erleichtert.
5. Wenn Ihre App auf eine AWS Ressource zugreift, übergeben Sie die Credentials Provider-Instance an das Client-Objekt, das temporäre Sicherheitsanmeldedaten an den Client weitergibt. Die Berechtigungen für die Anmeldeinformationen basieren auf der Rolle bzw. den Rollen, die Sie zuvor definiert haben.

Weitere Informationen finden Sie unter:

- Melden Sie sich in der AWS Amplify Framework-Dokumentation an (Android).
- Melden Sie sich in der AWS Amplify Framework-Dokumentation an (iOS).

Verwenden von OIDC-Verbund-API-Vorgängen für mobile Apps

Die besten Ergebnisse erzielen Sie, wenn Sie Amazon Cognito als Identitätsbroker für fast alle OIDC-Verbundszenarien verwenden. Amazon Cognito ist einfach zu bedienen und bietet zusätzliche Funktionen wie anonymen (nicht authentifizierten) Zugriff und die Synchronisierung von Benutzerdaten über Geräte und Anbieter hinweg. Wenn Sie jedoch bereits eine App erstellt haben, die den OIDC-Verbund verwendet, indem Sie die `AssumeRoleWithWebIdentity` API manuell aufrufen, können Sie sie weiterhin verwenden, und Ihre Apps funktionieren weiterhin einwandfrei.

Der Prozess für die Verwendung des OIDC-Verbunds ohne Amazon Cognito folgt dieser allgemeinen Gliederung:

1. Melden Sie sich als Entwickler über den externen Identitätsanbieter (Identity Provider, IdP) an und konfigurieren Sie Ihre App mit dem Identitätsanbieter, der Ihnen eine eindeutige ID für Ihre App übermittelt. (Verschiedene IdPs verwenden unterschiedliche Terminologie für diesen Prozess. In dieser Gliederung wird der Begriff „Konfigurieren“ für den Prozess der Identifizierung Ihrer App mit dem IdP verwendet.) Jeder IdP gibt dir eine App-ID, die für diesen IdP eindeutig ist. Wenn du also dieselbe App mit mehreren konfigurierst IdPs, hat deine App mehrere App-IDs. Sie können mehrere Apps mit jedem Anbieter konfigurieren.

   Die folgenden externen Links bieten Informationen zur Verwendung einiger der häufig verwendeten Identitätsanbieter ()IdPs:

   - Login with Amazon-Entwicklerzentrum
   - Add Facebook Login to Your App or Website auf der Facebook-Entwickler-Website.
   - Using OAuth 2.0 for Login (OpenID Connect) auf der Google-Entwickler-Website.

   > **Important**
   >
   > Wenn Sie einen OIDC-Identitätsanbieter von Google, Facebook oder Amazon Cognito verwenden, erstellen Sie keinen separaten IAM-Identitätsanbieter in der. AWS Management Console AWS hat diese OIDC-Identitätsanbieter integriert und stehen Ihnen zur Verfügung. Überspringen Sie den folgenden Schritt, und wechseln Sie direkt zum Erstellen neuer Rollen mit Ihrem Identitätsanbieter.

2. Wenn Sie einen anderen IdentiIdP als Google, Facebook oder Amazon Cognito verwenden, der mit OIDC kompatibel ist, erstellen Sie dafür eine IAM-Identitätsanbieter-Entität.
3. Erstellen Sie in IAM eine oder mehrere Rollen. Definieren Sie für jede Rolle, wer die Rolle übernehmen kann (die Vertrauensrichtlinie), und welche Berechtigungen die Benutzer der

App haben sollen (die Berechtigungsrichtlinie). In der Regel erstellen Sie eine Rolle für jeden Identitätsanbieter, den die Anwendung unterstützt. Sie können beispielsweise eine Rolle erstellen, die von einer Anwendung übernommen wird, wenn sich der Benutzer über Login with Amazon anmeldet, eine zweite Rolle für die gleiche Anwendung, in der der Benutzer sich mit Facebook anmeldet, und eine dritte Rolle für die Anwendung, in der sich der Benutzer mit Google angemeldet. Für die Vertrauensbeziehung geben Sie den Identitätsanbieter (wie Amazon.com) als `Principal` (die vertrauenswürdigen Entität) an und schließen eine `Condition` ein, die der vom Identitätsanbieter zugewiesenen App-ID entspricht. Beispiele von Rollen für verschiedene Anbieter sind in Erstellen von Rollen für externe Identitätsanbieter (Verbund) beschrieben.

4. Authentifizieren Sie Ihre Benutzer in der Anwendung mit dem Identitätsanbieter. Die Einzelheiten zur Vorgehensweise unterscheiden sich sowohl nach Maßgabe des verwendeten Identitätsanbieters (Login with Amazon, Facebook oder Google) als auch der Plattform, auf der die App ausgeführt wird. Beispielsweise kann sich die Authentifizierungsmethode einer Android-App von der einer iOS-App oder einer JavaScript basierten Web-App unterscheiden.

   Wenn der Benutzer nicht bereits angemeldet haben, zeigt der Identitätsanbieter in der Regel eine Anmeldeseite an. Nachdem der Identitätsanbieter den Benutzer authentifiziert hat, gibt er einen Authentifizierungs-Token mit Informationen über den Benutzer an Ihre App zurück. Die Informationen hängt davon ab, was der Identitätsanbieter bereitstellt und welche Informationen der Benutzer weitergeben möchte. Sie können diese Informationen in Ihrer App verwenden.

5. Führen Sie in Ihrer App einen nicht signierten Aufruf an die `AssumeRoleWithWebIdentity`-Aktion durch, um temporäre Sicherheitsanmeldeinformationen anzufordern. In der Anfrage übergeben Sie das Authentifizierungstoken des IdP und geben den Amazon-Ressourcennamen (ARN) für die IAM-Rolle an, die Sie für diesen IdP erstellt haben. AWS überprüft, ob das Token vertrauenswürdig und gültig ist, und gibt in diesem Fall temporäre Sicherheitsanmeldeinformationen an Ihre App zurück, die über die Berechtigungen für die Rolle verfügen, die Sie in der Anfrage angeben. Die Antwort umfasst auch Metadaten über den Benutzer vom Identitätsanbieter, wie z. B. die eindeutige Benutzer-ID, die der Identitätsanbieter dem Benutzer zuweist.

6. Mithilfe der temporären Sicherheitsanmeldedaten aus der `AssumeRoleWithWebIdentity` Antwort stellt Ihre App signierte Anfragen an AWS API-Operationen. Die Benutzer-ID-Informationen des IdP können Benutzer in Ihrer App unterscheiden. Sie können beispielsweise Objekte in Amazon S3 S3-Ordnern ablegen, die die Benutzer-ID als Präfixe oder Suffixe enthalten. Dadurch können Sie Richtlinien zur Zugriffskontrolle erstellen, die den Ordner sperren, sodass nur der Benutzer mit der entsprechenden ID darauf zugreifen kann. Weitere Informationen finden Sie unter AWS STS Prinzipale für föderierte Benutzersitzungen.

7. Ihre App sollte die temporären Sicherheitsanmeldeinformationen zwischenspeichern, sodass Sie nicht jedes Mal, wenn die App eine Anforderung an AWS machen muss, neue abrufen müssen. Die Anmeldeinformationen sind standardmäßig eine Stunde gültig. Wenn die Anmeldeinformationen abgelaufen sind (oder davor), rufen Sie `AssumeRoleWithWebIdentity` erneut auf, um einen neuen Satz an temporären Sicherheitsanmeldeinformationen abzurufen. Abhängig vom Identitätsanbieter und der Vorgehensweise beim Verwalten seiner Token müssen Sie möglicherweise das Token aktualisieren, bevor Sie einen neuen Aufruf an `AssumeRoleWithWebIdentity` machen, da die Token des Identitätsanbieters normalerweise nach einer bestimmten Zeit ablaufen. Wenn Sie das AWS SDK for iOS oder das AWS SDK for Android verwenden, können Sie die CredentialsProviderAmazonSTS-Aktion verwenden, die die temporären IAM-Anmeldeinformationen verwaltet und sie bei Bedarf aktualisiert.

OIDC-Föderation

Stellen Sie sich vor, Sie erstellen eine Anwendung, die auf AWS Ressourcen zugreift, z. B. GitHub Aktionen, die Workflows für den Zugriff auf Amazon S3 und DynamoDB verwendet.

Wenn Sie diese Workflows verwenden, stellen Sie Anfragen an AWS Dienste, die mit einem AWS Zugriffsschlüssel signiert werden müssen. Wir empfehlen jedoch dringend, AWS Anmeldeinformationen nicht langfristig in externen Anwendungen zu speichern AWS. Konfigurieren Sie Ihre Anwendungen stattdessen mithilfe des OIDC-Verbunds so, dass temporäre AWS Sicherheitsanmeldedaten bei Bedarf dynamisch angefordert werden. Die bereitgestellten temporären Anmeldeinformationen sind einer AWS Rolle zugeordnet, die nur über Berechtigungen verfügt, die zur Ausführung der für die Anwendung erforderlichen Aufgaben erforderlich sind.

Mit dem OIDC-Verbund müssen Sie keinen benutzerdefinierten Anmeldecode erstellen oder Ihre eigenen Benutzeridentitäten verwalten. Stattdessen können Sie OIDC in Anwendungen wie GitHub Actions oder jedem anderen OpenID Connect (OIDC) -kompatiblen IdP für die Authentifizierung verwenden. AWS Sie erhalten ein Authentifizierungstoken, das als JSON Web Token (JWT) bezeichnet wird, und tauschen dieses Token dann gegen temporäre Sicherheitsanmeldedaten aus, die einer IAM-Rolle mit der Berechtigung zur Nutzung bestimmter Ressourcen in AWS Ihrem zugeordnet sind. AWS-Konto Die Verwendung eines IdP hilft Ihnen dabei, Ihre AWS-Konto Sicherheit zu gewährleisten, da Sie keine langfristigen Sicherheitsnachweise in Ihre Anwendung einbetten und verteilen müssen.

Für die meisten Szenarien empfehlen wir die Verwendung von Amazon Cognito, da es als Identity Broker fungiert und einen Großteil der Föderationsarbeit für Sie übernimmt. Weitere Informationen finden Sie im folgenden Abschnitt, Verwenden von Amazon Cognito für mobile Apps.

> **Note**
>
> Von OpenID Connect (OIDC) -Identitätsanbietern ausgegebene JSON-Web-Tokens (JWTs) enthalten im `exp` Anspruch eine Ablaufzeit, die angibt, wann das Token abläuft. IAM bietet ein Zeitfenster von fünf Minuten nach der im JWT angegebenen Ablaufzeit, um Zeitversatz zu berücksichtigen, wie es der OpenID Connect (OIDC) Core 1.0-Standard zulässt. Das bedeutet, dass OIDC-JWTs, die nach Ablauf der Ablaufzeit, aber innerhalb dieses Fünf-Minuten-Zeitfensters von IAM empfangen werden, zur weiteren Auswertung und Verarbeitung akzeptiert werden.

Themen

Erstellen Sie einen OpenID Connect (OIDC) -Identitätsanbieter in IAM

IAM-OIDC-Identitätsanbieter sind Entitäten in IAM, die einen externen Identitätsanbieter (IdP)-Service beschreiben, der den OpenID Connect (OIDC)-Standard unterstützt, wie Google oder Salesforce. Sie verwenden einen IAM-OIDC-Identitätsanbieter, wenn Sie eine Vertrauensstellung zwischen einem OIDC-kompatiblen Identitätsanbieter und Ihrem AWS-Konto einrichten möchten. Dies ist nützlich, wenn Sie eine mobile App oder Webanwendung erstellen, für die Zugriff auf AWS Ressourcen erforderlich ist, Sie aber keinen benutzerdefinierten Anmeldecode erstellen oder Ihre eigenen Benutzeridentitäten verwalten möchten. Weitere Informationen zu diesem Szenario finden Sie unter the section called “OIDC-Föderation”.

Sie können einen IAM-OIDC-Identitätsanbieter mithilfe der AWS Management Console Tools für Windows PowerShell oder der AWS Command Line Interface IAM-API erstellen und verwalten.

Nachdem Sie einen IAM OIDC-Identitätsanbieter erstellt haben, müssen Sie eine oder mehrere IAM-Rollen erstellen. Eine Rolle ist eine Identität AWS , die keine eigenen Anmeldeinformationen hat (wie dies bei einem Benutzer der Fall ist). Aber in diesem Kontext ist eine Rolle dynamisch einem Verbundbenutzer zugewiesen, der vom Identitätsanbieter der Organisation authentifiziert wird. Die Rolle ermöglicht es dem Identitätsanbieter Ihres Unternehmens, temporäre Sicherheitsanmeldeinformationen für den Zugriff auf AWS anzufordern. Die der Rolle zugewiesenen Richtlinien legen fest, was die Verbundbenutzer tun dürfen. AWS Informationen zum Erstellen einer

Rolle für einen Drittanbieter-Identitätsanbieter finden Sie unter Erstellen von Rollen für externe Identitätsanbieter (Verbund).

> **Important**
>
> Wenn Sie identitätsbasierte Richtlinien für Aktionen konfigurieren, die `oidc-provider`-Ressourcen unterstützen, bewertet IAM die vollständige URL des OIDC-Identitätsanbieters, einschließlich aller angegebenen Pfade. Wenn Ihre OIDC-Identitätsanbieter-URL über einen Pfad verfügt, müssen Sie diesen Pfad in der `oidc-provider`-ARN als `Resource`-Elementwert angeben. Sie haben auch die Möglichkeit, einen Schrägstrich und einen Platzhalter (`/*`) an die URL-Domain anzuhängen oder Platzhalterzeichen (`*` und `?`) an einer beliebigen Stelle im URL-Pfad zu verwenden. Wenn die URL des OIDC-Identitätsanbieters in der Anforderung nicht mit dem im `Resource`-Element der Richtlinie festgelegten Wert übereinstimmt, schlägt die Anforderung fehl.

Informationen zur Behebung häufiger Probleme mit dem IAM-OIDC-Verbund finden Sie unter Beheben von Fehlern im Zusammenhang mit OIDC auf re:POST. AWS

Themen

- Voraussetzungen: Bestätigen Sie die Konfiguration Ihres Identitätsanbieters
- Erstellen und Verwalten eines OIDC-Anbieters (Konsole)
- Erstellen und Verwalten eines IAM-OIDC-Identitätsanbieters (AWS CLI)
- Einen OIDC Identity Provider (API) erstellen und verwalten AWS

Voraussetzungen: Bestätigen Sie die Konfiguration Ihres Identitätsanbieters

Bevor Sie einen IAM-OIDC-Identitätsanbieter erstellen können, benötigen Sie die folgenden Informationen von Ihrem IdP. Weitere Informationen zum Abrufen von Informationen zur OIDC-Provider-Konfiguration finden Sie in der Dokumentation für Ihren IdP.

1. Ermitteln Sie die öffentlich verfügbare URL Ihres OIDC-Identitätsanbieters. Die URL muss mit https:// beginnen. Gemäß dem OIDC-Standard sind Pfadkomponenten zulässig, Abfrageparameter jedoch nicht. In der Regel besteht die URL nur aus einem Hostnamen wie https://server.example.org oder https://example.com. Die URL darf keine Portnummer enthalten.
2. Fügen Sie /.well-known/openid-configuration am Ende der URL Ihres OIDC-Identitätsanbieters hinzu, um das öffentlich verfügbare Konfigurationsdokument und die Metadaten des

Anbieters zu sehen. Sie benötigen ein Discovery-Dokument im JSON-Format mit dem Konfigurationsdokument und den Metadaten des Anbieters, das über die URL des OpenID Connect-Provider-Discovery-Endpunkts abgerufen werden kann.

3. Vergewissern Sie sich, dass die folgenden Werte in den Konfigurationsinformationen Ihres Anbieters enthalten sind. Wenn in Ihrer Openid-Konfiguration eines dieser Felder fehlt, müssen Sie Ihr Discovery-Dokument aktualisieren. Dieser Vorgang kann je nach Ihrem Identitätsanbieter variieren. Folgen Sie daher der Dokumentation Ihres IdP, um diese Aufgabe abzuschließen.

   - Aussteller: Die URL für Ihre Domain.
   - jwks_uri: Der JWKS-Endpunkt (JSON Web Key Set), an dem IAM Ihre öffentlichen Schlüssel abruft. Ihr Identitätsanbieter muss einen JSON Web Key Set (JWKS) -Endpunkt in die Openid-Konfiguration aufnehmen. Dieser URI definiert, wo Sie Ihre öffentlichen Schlüssel erhalten, die zur Überprüfung der signierten Token von Ihrem Identitätsanbieter verwendet werden.
   - claims_supported: Informationen über den Benutzer, anhand derer Sie sicherstellen können, dass die OIDC-Authentifizierungsantworten Ihres IdP die erforderlichen Attribute enthalten, die in IAM-Richtlinien AWS verwendet werden, um die Berechtigungen für Verbundbenutzer zu überprüfen. Eine Liste der IAM-Bedingungsschlüssel, die für Ansprüche verwendet werden können, finden Sie unter. Verfügbare Schlüssel für AWS den OIDC-Verbund
     - aud: Sie müssen in JSON Web Tokens (JWTs) ermitteln, welchen Wert Ihre IdP-Probleme für die Zielgruppe beanspruchen. Der Zielgruppenanspruch (aud) ist anwendungsspezifisch und identifiziert die beabsichtigten Empfänger des Tokens. Wenn Sie eine Mobil- oder Web-App bei einem OpenID Connect-Anbieter registrieren, richtet dieser eine Client-ID ein, die die Anwendung identifiziert. Die Client-ID ist eine eindeutige Kennung für Ihre App, die im Aud-Authentifizierungsantrag übergeben wird. Der Aud-Anspruch muss mit dem Wert Audience übereinstimmen, den Sie bei der Erstellung Ihres IAM OIDC-Identitätsanbieters angegeben haben.
     - iat: Anträge müssen einen Wert enthalten`iat`, der den Zeitpunkt angibt, zu dem das ID-Token ausgestellt wurde.
     - iss: Die URL des Identitätsanbieters. Die URL muss mit https://beginnen und der Provider-URL entsprechen, die IAM bereitgestellt wurde. Gemäß dem OIDC-Standard sind Pfadkomponenten zulässig, Abfrageparameter jedoch nicht. In der Regel besteht die URL nur aus einem Hostnamen wie https://server.example.org oder https://example.com. Die URL darf keine Portnummer enthalten.
   - response_types_supported: id_token
   - subject_types_supported: öffentlich

- id_token_signing_alg_values_supported: RS256

> **Note**
>
> Sie können zusätzliche Ansprüche wie benutzerdefinierte im Beispiel unten angeben; der Anspruch wird jedoch ignoriert. AWS STS

```
{
  "issuer": "https://example-domain.com",
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "custom"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256"
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

Erstellen und Verwalten eines OIDC-Anbieters (Konsole)

Folgen Sie diesen Anweisungen, um einen IAM OIDC-Identitätsanbieter im AWS Management Console zu erstellen und zu verwalten.

> **Important**
>
> Wenn Sie einen IAM-OIDC-Identitätsanbieter von Google, Facebook oder Amazon Cognito verwenden, erstellen Sie mit diesem Verfahren keinen separaten IAM-Identitätsanbieter. Diese OIDC-Identitätsanbieter sind bereits integriert AWS und stehen Ihnen zur Verfügung.

Folgen Sie stattdessen den Schritten zum Erstellen neuer Rollen für Ihren Identitätsanbieter, siehe Eine Rolle für den OpenID Connect-Verbund erstellen (Konsole).

So erstellen Sie einen IAM OIDC-Identitätsanbieter (Konsole)

1. Bevor Sie einen IAM-OIDC-Identitätsanbieter erstellen, müssen Sie Ihre Anwendung beim Identitätsanbieter registrieren, um eine Client-ID zu erhalten. Die Client-ID (auch als Zielgruppe bezeichnet) ist ein eindeutiger Bezeichner für Ihre App, der ausgestellt wird, wenn Sie Ihre App beim Identitätsanbieter registrieren. Weitere Informationen über den Erhalt einer Client-ID finden Sie in der Dokumentation Ihres Identitätsanbieters.

   > **Note**
   >
   > AWS sichert die Kommunikation mit einigen OIDC-Identitätsanbietern (IdPs) über unsere Bibliothek vertrauenswürdiger Stammzertifizierungsstellen (CAs), anstatt einen Zertifikatsfingerabdruck zur Überprüfung Ihres IdP-Serverzertifikats zu verwenden. In diesen Fällen bleibt Ihr alter Fingerabdruck in Ihrer Konfiguration erhalten, wird aber nicht mehr zur Validierung verwendet. Zu diesen OIDC IdPs gehören Auth0,,, Google und solche GitHub GitLab, die einen Amazon S3 S3-Bucket verwenden, um einen JSON Web Key Set (JWKS) -Endpunkt zu hosten.

2. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
3. Wählen Sie im Navigationsbereich Identitätsanbieter und dann Anbieter hinzufügen.
4. Wählen Sie für Anbieter konfigurieren die Option OpenID Connect.
5. Geben Sie für Provider URL die URL des Identitätsanbieters ein. Die URL muss den folgenden Einschränkungen entsprechen:

   - Die URL berücksichtigt Groß- und Kleinschreibung.
   - Die URL muss mit **`https://`** beginnen.
   - Die URL darf keine Portnummer enthalten.
   - In Ihrem muss AWS-Konto jeder IAM-OIDC-Identitätsanbieter eine eindeutige URL verwenden. Wenn Sie versuchen, eine URL einzureichen, die bereits für einen OpenID Connect-Anbieter in der verwendet wurde AWS-Konto, erhalten Sie eine Fehlermeldung.

6. Geben Sie für Audience die Client-ID der Anwendung ein, die Sie beim IdP registriert und in Step 1 der Sie empfangen haben und an AWS die Sie Anfragen stellen. Wenn Sie zusätzliche Client-

IDs (auch als Zielgruppen bekannt) für diesen Identitätsanbieter haben, können Sie sie zu einem späteren Zeitpunkt auf der Detailseite des Anbieters hinzufügen.

> **Note**
>
> Wenn Ihr IdP-JWT-Token den `azp` Anspruch enthält, geben Sie diesen Wert als Zielgruppenwert ein.

7. (Optional) Für Tags hinzufügen können Sie Schlüssel-Wert-Paare hinzufügen, um Ihre Tags leichter identifizieren und organisieren zu können. IdPs Sie können auch Tags verwenden, um den Zugriff auf AWS -Ressourcen zu steuern. Weitere Informationen zum Markieren von IAM-OIDC-Identitätsanbietern finden Sie unter Markieren von OpenID-Connect(OIDC)-Identitätsanbietern. Wählen Sie Add tag. Geben Sie Werte für jedes Tag-Schlüsselwertpaar ein.
8. Verifizieren Sie die angegebenen Informationen. Wenn Sie fertig sind, wählen Sie Anbieter hinzufügen. IAM versucht, den obersten CA-Fingerabdruck des OIDC-IdP-Serverzertifikats abzurufen und zu verwenden, um den IAM-OIDC-Identitätsanbieter zu erstellen.

> **Note**
>
> Die Zertifikatskette des OIDC-Identitätsanbieters muss mit der Domain- oder Aussteller-URL beginnen, dann mit dem Zwischenzertifikat und mit dem Stammzertifikat enden. Wenn die Reihenfolge der Zertifikatskette unterschiedlich ist oder doppelte oder zusätzliche Zertifikate enthält, erhalten Sie eine Fehlermeldung, dass die Signatur nicht übereinstimmt, und STS kann das JSON Web Token (JWT) nicht validieren. Korrigieren Sie die Reihenfolge der Zertifikate in der Kette, die vom Server zurückgegeben wurden, um den Fehler zu beheben. Weitere Informationen zu Zertifikatskettenstandards finden Sie unter certificate_list in RFC 5246 auf der Website der RFC-Serie.

9. Weisen Sie Ihrem Identitätsanbieter eine IAM-Rolle zu, um externen Benutzeridentitäten, die von Ihrem Identitätsanbieter verwaltet werden, Zugriff auf Ressourcen in Ihrem Konto zu gewähren. AWS Weitere Informationen zum Erstellen von Rollen für den Identitätsverbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter (Verbund).

> **Note**
>
> OIDC, das in einer Rollenvertrauensrichtlinie IdPs verwendet wird, muss sich in demselben Konto befinden wie die Rolle, die der Rolle vertraut.

So fügen Sie einen Fingerabdruck für einen IAM-OIDC-Identitätsanbieter (Konsole) hinzu oder entfernen ihn

> **Note**
>
> AWS sichert die Kommunikation mit einigen OIDC-Identitätsanbietern (IdPs) über unsere Bibliothek vertrauenswürdiger Stammzertifizierungsstellen (CAs), anstatt einen Zertifikatsfingerabdruck zur Überprüfung Ihres IdP-Serverzertifikats zu verwenden. In diesen Fällen bleibt Ihr alter Fingerabdruck in Ihrer Konfiguration erhalten, wird aber nicht mehr zur Validierung verwendet. Zu diesen OIDC IdPs gehören Auth0,,, Google und solche GitHub GitLab, die einen Amazon S3 S3-Bucket verwenden, um einen JSON Web Key Set (JWKS) - Endpunkt zu hosten.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Identitätsanbieter. Wählen Sie dann den Namen des IAM-Identitätsanbieters aus, den Sie aktualisieren möchten.
3. Wählen Sie im Abschnitt Fingerabdrücke die Option Verwalten aus. Um einen neuen Fingerabdruck einzugeben, wählen Sie Fingerabdruck hinzufügen. Wählen Sie zum Entfernen eines Fingerabdrucks neben dem Fingerabdruck, den Sie entfernen möchten, Entfernen.

   > **Note**
   >
   > Ein IAM OIDC-Identitätsanbieter muss mindestens einen und kann maximal fünf Thumbprints haben.

   Wählen Sie abschließend Änderungen speichern.

So fügen Sie eine Zielgruppe für einen IAM OIDC-Identitätsanbieter (Konsole) hinzu

1. Wählen Sie im Navigationsbereich die Option Identitätsanbieter und dann den Namen des IAM-Identitätsanbieters, den Sie aktualisieren möchten.
2. Wählen Sie im Abschnitt Zielgruppen die Option Aktionen und wählen Sie Zielgruppe hinzufügen.
3. Geben Sie die Client-ID der Anwendung ein, die Sie beim IdP registriert und in Step 1 der Sie empfangen haben und an AWS die Anfragen gestellt werden. Wählen Sie dann Zielgruppen hinzufügen.

> **Note**
>
> Ein IAM OIDC-Identitätsanbieter muss mindestens eine und kann maximal 100 Zielgruppen haben.

So entfernen Sie eine Zielgruppe für einen IAM OIDC-Identitätsanbieter (Konsole)

1. Wählen Sie im Navigationsbereich die Option Identitätsanbieter und dann den Namen des IAM-Identitätsanbieters, den Sie aktualisieren möchten.
2. Wählen Sie im Abschnitt Zielgruppen das Optionsfeld neben der Zielgruppe aus, die Sie entfernen möchten, und wählen Sie dann Aktionen.
3. Wählen Sie Zielgruppe entfernen. Ein neues Fenster wird geöffnet.
4. Wenn Sie eine Zielgruppe entfernen, können Identitäten, die mit der Zielgruppe verbunden sind, keine mit der Zielgruppe verbundenen Rollen annehmen. Lesen Sie im Fenster die Warnung und bestätigen Sie, dass Sie die Zielgruppe entfernen möchten, indem Sie das Wort `remove` in das Feld eingeben.
5. Wählen Sie Entfernen, um die Zielgruppe zu entfernen.

So löschen Sie einen IAM OIDC-Identitätsanbieter (Konsole)

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Identitätsanbieter.
3. Aktivieren Sie das Kontrollkästchen neben dem IAM-Identitätsanbieter, den Sie löschen möchten. Ein neues Fenster wird geöffnet.
4. Bestätigen Sie, dass Sie den Anbieter löschen möchten, indem Sie das Wort `delete` in das Feld eingeben. Wählen Sie dann Löschen.

Erstellen und Verwalten eines IAM-OIDC-Identitätsanbieters (AWS CLI)

Sie können die folgenden AWS CLI Befehle verwenden, um IAM OIDC-Identitätsanbieter zu erstellen und zu verwalten.

So erstellen Sie einen IAM-OIDC-Identitätsanbieter (AWS CLI)

1. (Optional) Führen Sie zum Abrufen einer Liste aller IAM-OIDC-Identitätsanbieter im AWS -Konto den folgenden Befehl aus:
   - `aws iam list-open-id-connect-providers`
2. Führen Sie zum Erstellen eines neuen IAM-OIDC-Identitätsanbieters den folgenden Befehl aus:
   - `aws iam create-open-id-connect-provider`

So aktualisieren Sie die Liste der Serverzertifikat-Thumbprints für einen vorhandenen IAM-OIDC-Identitätsanbieter (AWS CLI)

- Führen Sie zum Aktualisieren der Liste der Serverzertifikat-Thumbprints für einen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
  - `aws iam update-open-id-connect-provider-thumbprint`

Markieren eines bestehenden IAM-OIDC-Identitätsanbieters (AWS CLI)

- Führen Sie zum Markieren eines bestehenden IAM-OIDC-Identitätsanbieters den folgenden Befehl aus:
  - `aws iam tag-open-id-connect-provider`

Auflisten von Tags für einen bestehenden IAM-OIDC-Identitätsanbieter (AWS CLI)

- Führen Sie zum Auflisten von Tags für einen vorhandenen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
  - `aws iam list-open-id-connect-provider-tags`

Entfernen von Tags auf einem IAM-OIDC-Identitätsanbieter (AWS CLI)

- Führen Sie zum Entfernen von Tags auf einem vorhandenen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
  - `aws iam untag-open-id-connect-provider`

So fügen Sie eine Client-ID von einem vorhandenen IAM-OIDC-Identitätsanbieter hinzu oder entfernen sie (AWS CLI)

1. (Optional) Führen Sie zum Abrufen einer Liste aller IAM-OIDC-Identitätsanbieter im AWS -Konto den folgenden Befehl aus:
   - `aws iam list-open-id-connect-providers`
2. (Optional) Führen Sie zum Abrufen detaillierter Informationen über einen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
   - `aws iam get-open-id-connect-provider`
3. Führen Sie zum Hinzufügen einer neuen Client-ID zu einem vorhandenen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
   - `aws iam add-client-id-to-open-id-connect-provider`
4. Führen Sie zum Entfernen eines Clients von einem vorhandenen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
   - `aws iam remove-client-id-from-open-id-connect-provider`

So löschen Sie einen IAM-OIDC-Identitätsanbieter (AWS CLI)

1. (Optional) Führen Sie zum Abrufen einer Liste aller IAM-OIDC-Identitätsanbieter im AWS -Konto den folgenden Befehl aus:
   - `aws iam list-open-id-connect-providers`
2. (Optional) Führen Sie zum Abrufen detaillierter Informationen über einen IAM-OIDC-Identitätsanbieter den folgenden Befehl aus:
   - `aws iam get-open-id-connect-provider`
3. Führen Sie zum Löschen eines IAM-OIDC-Identitätsanbieters den folgenden Befehl aus:
   - `aws iam delete-open-id-connect-provider`

Einen OIDC Identity Provider (API) erstellen und verwalten AWS

Mit den folgenden IAM-API-Befehlen können Sie OIDC-Anbieter erstellen und verwalten.

So erstellen Sie einen IAM-OIDC-Identitätsanbieter (API)AWS

1. (Optional) Rufen Sie zum Abrufen einer Liste aller IAM-OIDC-Identitätsanbieter im AWS -Konto die folgende Operation auf:
   - `ListOpenIDConnectProviders`
2. Rufen Sie zum Erstellen eines neuen IAM-OIDC-Identitätsanbieters die folgende Operation auf:
   - `CreateOpenIDConnectProvider`

Um die Liste der Fingerabdrücke von Serverzertifikaten für einen vorhandenen IAM-OIDC-Identitätsanbieter (API) zu aktualisieren AWS

- Rufen Sie zum Aktualisieren der Liste der Serverzertifikat-Thumbprints für einen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
  - `UpdateOpenIDConnectProviderThumbprint`

Um einen vorhandenen IAM-OIDC-Identitätsanbieter (API) zu taggen AWS

- Rufen Sie zum Markieren eines bestehenden IAM-OIDC-Identitätsanbieters die folgende Operation auf:
  - `TagOpenIDConnectProvider`

Um Tags für einen vorhandenen IAM OIDC-Identitätsanbieter (API) aufzulisten AWS

- Rufen Sie zum Auflisten von Tags für einen vorhandenen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
  - `ListOpenIDConnectProviderTags`

Um Tags auf einem vorhandenen IAM OIDC-Identitätsanbieter (API) zu entfernen AWS

- Rufen Sie zum Entfernen von Tags auf einem vorhandenen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
  - `UntagOpenIDConnectProvider`

So fügen Sie eine Client-ID von einem vorhandenen IAM-OIDC-Identitätsanbieter hinzu oder entfernen sie (AWS -API)

1. (Optional) Rufen Sie zum Abrufen einer Liste aller IAM-OIDC-Identitätsanbieter im AWS -Konto die folgende Operation auf:
   - `ListOpenIDConnectProviders`
2. (Optional) Rufen Sie zum Abrufen detaillierter Informationen über einen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
   - `GetOpenIDConnectProvider`
3. Rufen Sie zum Hinzufügen einer neuen Client-ID zu einem vorhandenen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
   - `AddClientIDToOpenIDConnectProvider`
4. Rufen Sie zum Entfernen einer Client-ID von einem vorhandenen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
   - `RemoveClientIDFromOpenIDConnectProvider`

Um einen IAM-OIDC-Identitätsanbieter (API) zu löschen AWS

1. (Optional) Rufen Sie zum Abrufen einer Liste aller IAM-OIDC-Identitätsanbieter im AWS -Konto die folgende Operation auf:
   - `ListOpenIDConnectProviders`
2. (Optional) Rufen Sie zum Abrufen detaillierter Informationen über einen IAM-OIDC-Identitätsanbieter die folgende Operation auf:
   - `GetOpenIDConnectProvider`
3. Rufen Sie zum Löschen eines IAM-OIDC-Identitätsanbieters die folgende Operation auf:
   - `DeleteOpenIDConnectProvider`

Besorgen Sie sich den Fingerabdruck für einen OpenID Connect-Identitätsanbieter

Wenn Sie einen OpenID Connect (OIDC) -Identitätsanbieter in IAM erstellen, benötigt IAM den Fingerabdruck der obersten Zwischenzertifizierungsstelle (CA), die das vom externen

Identitätsanbieter (IdP) verwendete Zertifikat signiert hat. Bei dem Thumbprint handelt es sich um eine Signatur für das CA-Zertifikat, das für die Ausstellung des Zertifikats für den OIDC-kompatiblen Identitätsanbieter verwendet wurde. Wenn Sie einen IAM-OIDC-Identitätsanbieter erstellen, vertrauen Sie darauf, dass Identitäten, die von diesem IdP authentifiziert wurden, Zugriff auf Ihren haben. AWS-Konto Wenn Sie den Fingerabdruck des Zertifikats der Zertifizierungsstelle verwenden, vertrauen Sie jedem Zertifikat, das von dieser Zertifizierungsstelle ausgestellt wurde und denselben DNS-Namen hat wie das registrierte. Dadurch müssen die Vertrauensstellungen in den einzelnen Konten nicht aktualisiert werden, wenn Sie das Signaturzertifikat des Identitätsanbieters erneuern.

Important

In den meisten Fällen verwendet der Verbundserver zwei verschiedene Zertifikate:

- Die erste stellt eine HTTPS-Verbindung zwischen AWS und Ihrem IdP her. Dies sollte von einer bekannten öffentlichen Root-CA ausgestellt werden, z. B. AWS Certificate Manager Dadurch kann der Kunde die Zuverlässigkeit und den Status des Zertifikats überprüfen.
- Die zweite wird zum Verschlüsseln von Token verwendet und sollte von einer privaten oder öffentlichen Stamm-Zertifizierungsstelle signiert werden.

Sie können einen IAM-OIDC-Identitätsanbieter mit den AWS Command Line Interface Tools für Windows oder der PowerShell IAM-API erstellen. Wenn Sie diese Methoden verwenden, haben Sie die Möglichkeit, manuell einen Fingerabdruck bereitzustellen. Wenn Sie keinen Fingerabdruck angeben möchten, ruft IAM den obersten CA-Fingerabdruck der Zwischenstelle des OIDC-IdP-Serverzertifikats ab. Wenn Sie sich dafür entscheiden, einen Fingerabdruck hinzuzufügen, müssen Sie den Fingerabdruck manuell abrufen und an ihn weiterleiten. AWS

Wenn Sie mit der IAM-Konsole einen OIDC-Identitätsanbieter erstellen, versucht IAM, den obersten Zwischenzertifizierungsstellen-Fingerabdruck des OIDC-IdP-Serverzertifikats für Sie abzurufen.

Wir empfehlen Ihnen, den Fingerabdruck für Ihren OIDC-IdP auch manuell abzurufen und zu überprüfen, ob IAM den richtigen Fingerabdruck abgerufen hat. Weitere Informationen zum Abrufen von Fingerabdrücken von Zertifikaten finden Sie in den folgenden Abschnitten.

Note

AWS sichert die Kommunikation mit einigen OIDC-Identitätsanbietern (IdPs) über unsere Bibliothek vertrauenswürdiger Stammzertifizierungsstellen (CAs), anstatt einen

Zertifikatsfingerabdruck zur Überprüfung Ihres IdP-Serverzertifikats zu verwenden. In diesen Fällen bleibt Ihr alter Fingerabdruck in Ihrer Konfiguration erhalten, wird aber nicht mehr zur Validierung verwendet. Zu diesen OIDC IdPs gehören Auth0,,, Google und solche GitHub GitLab, die einen Amazon S3 S3-Bucket verwenden, um einen JSON Web Key Set (JWKS) - Endpunkt zu hosten.

Informationen zur Behebung häufiger Probleme mit dem IAM-OIDC-Verbund finden Sie unter Beheben von Fehlern im Zusammenhang mit OIDC auf re:POST. AWS

Besorgen Sie sich den Fingerabdruck des Zertifikats

Sie verwenden einen Webbrowser und das OpenSSL-Befehlszeilentool, um den Fingerabdruck des Zertifikats für einen OIDC-Anbieter abzurufen. Sie müssen den Fingerabdruck des Zertifikats jedoch nicht manuell abrufen, um einen IAM-OIDC-Identitätsanbieter zu erstellen. Sie können das folgende Verfahren verwenden, um den Fingerabdruck des Zertifikats Ihres OIDC-Anbieters abzurufen.

So rufen Sie den Thumbprint eines IAM-OIDC-Identitätsanbieters ab

1. Bevor Sie den Thumbprint eines IAM-OIDC-Identitätsanbieters abrufen können, müssen Sie sich das OpenSSL-Befehlszeilen-Tool herunterladen. Mit diesem Tool laden Sie dann die Zertifikatskette des OIDC-Identitätsanbieters herunter und erstellen einen Thumbprint des letzten Zertifikats in der Zertifikatskette. Eine Anleitung zum Installieren und Konfigurieren von OpenSSL finden Sie unter Installieren von OpenSSL und Konfigurieren von OpenSSL.
2. Beginnen Sie mit der URL des OIDC-Identitätsanbieters (z. B. `https://server.example.com`) und fügen Sie `/.well-known/openid-configuration` hinzu, um die URL für das Konfigurationsdokument des Identitätsanbieters wie folgt zu bilden:

   `https://`*`server.example.com`*`/.well-known/openid-configuration`

   Öffnen Sie diese URL in einem Webbrowser und ersetzen Sie dabei *`server.example.com`* durch den Servernamen Ihres Identitätsanbieters.
3. Verwenden Sie im angezeigten Dokument die Such-Feature Ihres Webbrowsers, um den Text `"jwks_uri"` zu finden. Direkt hinter dem Text `"jwks_uri"` finden Sie einen Doppelpunkt (:) gefolgt von einer URL. Kopieren Sie den vollständig qualifizierten Domainnamen der URL. Kopieren Sie weder `https://` noch Teile des Pfads hinter der Top-Level-Domain.

   ```
   {
   ```

```
 "issuer": "https://accounts.example.com",
 "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
 "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
 "token_endpoint": "https://oauth2.exampleapis.com/token",
 "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
 "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
 "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
...
```

4. Führen Sie mithilfe des OpenSSL-Befehlszeilen-Tools den folgenden Befehl. Ersetzen Sie *keys.example.com* durch den Domainnamen, den Sie unter Step 3 erhalten haben.

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. Scrollen Sie im Befehlszeilenfenster nach unten, bis Sie ein Zertifikat ähnlich wie im folgenden Beispiel finden. Wenn mehrere Zertifikate angezeigt werden, suchen Sie nach dem letzten angezeigten Zertifikat (am Ende der Befehlsausgabe). Dies enthält das Zertifikat der Zertifizierungsstelle in der Kette der Zertifizierungsstellen.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
 NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
 -----END CERTIFICATE-----
```

Kopieren Sie das Zertifikat (einschließlich der Zeilen `-----BEGIN CERTIFICATE-----` und `-----END CERTIFICATE-----`) und fügen es in einer Textdatei ein. Speichern Sie die Datei mit dem Dateinamen **`certificate.crt`**.

> **Note**
>
> Die Zertifikatskette des OIDC-Identitätsanbieters muss mit der Domain- oder Aussteller-URL beginnen, dann mit dem Zwischenzertifikat und mit dem Stammzertifikat enden. Wenn die Reihenfolge der Zertifikatskette unterschiedlich ist oder doppelte oder zusätzliche Zertifikate enthält, erhalten Sie einen Signaturfehler, und STS kann das JSON Web Token (JWT) nicht validieren. Korrigieren Sie die Reihenfolge der Zertifikate in der Kette, die vom Server zurückgegeben wurden, um den Fehler zu beheben. Weitere Informationen zu Zertifikatskettenstandards finden Sie unter certificate_list in RFC 5246 auf der Website der RFC-Serie.

6. Führen Sie mithilfe des OpenSSL-Befehlszeilen-Tools den folgenden Befehl.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

Im Befehlsfenster wird der Thumbprint des Zertifikats ähnlich wie im folgenden Beispiel angezeigt:

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Entfernen Sie den Doppelpunkt (:) aus dieser Zeichenfolge, um den endgültigen Thumbprint wie folgt zu erhalten:

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

7. Wenn Sie den IAM-OIDC-Identitätsanbieter mit den Tools für Windows oder der IAM-API erstellen AWS CLI, ist die Angabe eines PowerShell Fingerabdrucks optional. Wenn Sie bei der Erstellung keinen Fingerabdruck angeben möchten, ruft IAM den obersten CA-Fingerabdruck der Zwischenstelle des OIDC-IdP-Serverzertifikats ab. Nachdem der IAM-OIDC-Identitätsanbieter erstellt wurde, können Sie diesen Fingerabdruck mit dem von IAM abgerufenen Fingerabdruck vergleichen.

   Wenn Sie den IAM-OIDC-Identitätsanbieter in der IAM-Konsole erstellen, versucht die Konsole, den obersten Zwischen-CA-Fingerabdruck des OIDC-IdP-Serverzertifikats für Sie abzurufen. Sie können diesen Fingerabdruck mit dem von IAM abgerufenen Fingerabdruck vergleichen. Nachdem der IAM-OIDC-Identitätsanbieter erstellt wurde, können Sie den Fingerabdruck für den

IAM-OIDC-Identitätsanbieter auf der Registerkarte Endpunktverifizierung auf der Konsolenseite mit der Zusammenfassung des OIDC-Anbieters einsehen.

> **Important**
>
> Wenn der Fingerabdruck, den Sie erhalten haben, nicht mit dem übereinstimmt, den Sie in den Fingerabdruckdetails des IAM-OIDC-Identitätsanbieters sehen, sollten Sie den OIDC-Anbieter nicht verwenden. Stattdessen sollten Sie den erstellten OIDC-Anbieter löschen und nach einiger Zeit erneut versuchen, den OIDC-Anbieter zu erstellen. Stellen Sie sicher, dass die Fingerabdrücke übereinstimmen, bevor Sie den Anbieter verwenden. Wenn die Thumbprints auch beim zweiten Versuch nicht übereinstimmen, kontaktieren Sie über das IAM-Forum AWS.

Installieren von OpenSSL

Falls OpenSSL noch nicht installiert ist, befolgen Sie die Anleitungen in diesem Abschnitt.

So installieren Sie OpenSSL unter Linux und Unix

1. Wechseln Sie zu OpenSSL: Source, Tarballs(https://openssl.org/source/).
2. Laden Sie die aktuelle Quelldatei herunter und erstellen Sie das Paket.

So installieren Sie OpenSSL unter Windows

1. Unter OpenSSL: Binary Distributions (https://wiki.openssl.org/index.php/Binaries) finden Sie eine Liste von Sites, von denen Sie die Windows-Version installieren können.
2. Folgen Sie den Anweisungen auf der ausgewählten Site, um die Installation zu starten.
3. Wenn Sie aufgefordert werden, die Microsoft Visual C++ 2008 Redistributables zu installieren, und diese nicht bereits auf Ihrem System installiert sind, wählen Sie den für Ihre Umgebung geeigneten Download-Link. Folgen Sie den Anweisungen des Microsoft Visual C++ 2008 Redistributable Setup Wizard.

> **Note**
>
> Wenn Sie nicht sicher sind, ob Microsoft Visual C++ 2008 Redistributables bereits auf Ihrem System installiert ist, können Sie zunächst versuchen, OpenSSL zu installieren. Das OpenSSL Installationsprogramm zeigt eine Warnung an, wenn Microsoft Visual C++

2008 Redistributables noch nicht installiert ist. Installieren Sie die Architektur (32-Bit oder 64-Bit), die der installierten Version von OpenSSL entspricht.

4. Wählen Sie nach der Installation von Microsoft Visual C++ 2008 Redistributables die entsprechende Version der OpenSSL-Binärdateien für Ihre Umgebung aus und speichern Sie die Datei lokal. Starten Sie den OpenSSL-Setup-Assistenten.
5. Folgen Sie den Anweisungen des OpenSSL-Setup-Assistenten.

Konfigurieren von OpenSSL

Bevor Sie OpenSSL Befehle verwenden, müssen Sie das Betriebssystem so konfigurieren, dass es Informationen über den Speicherort enthält, an dem OpenSSL installiert ist.

So konfigurieren Sie OpenSSL unter Linux oder Unix

1. Setzen Sie in der Befehlszeile die Variable `OpenSSL_HOME` auf den Speicherort der OpenSSL-Installation:

```
$ export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

2. Legen Sie den Pfad für die OpenSSL Installation fest:

```
$ export PATH=$PATH:$OpenSSL_HOME/bin
```

> **Note**
>
> Alle Änderungen, die Sie mit dem Befehl `export` an Umgebungsvariablen vornehmen, sind nur für die aktuelle Sitzung gültig. Sie können dauerhafte Änderungen an den Umgebungsvariablen vornehmen, indem Sie diese in Ihrer Shell-Konfigurationsdatei festlegen. Weitere Informationen finden Sie in der Dokumentation für das Betriebssystem Ihrer Instance.

So konfigurieren Sie OpenSSL unter Windows

1. Öffnen Sie ein Befehlszeilenfenster.
2. Setzen Sie die `OpenSSL_HOME`-Variable auf den Ort der OpenSSL-Installation:

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Setzen Sie die `OpenSSL_CONF`-Variable auf den Speicherort der Konfigurationsdatei in Ihrer OpenSSL-Installation:

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Legen Sie den Pfad für die OpenSSL Installation fest:

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Alle Änderungen, die Sie in einem Eingabeaufforderungsfenster an Windows-Umgebungsvariablen vornehmen, sind nur für die aktuelle Befehlszeilensitzung gültig. Sie können dauerhafte Änderungen an den Umgebungsvariablen vornehmen, indem Sie sie als Systemeigenschaften festlegen. Die genauen Verfahren hängen davon ab, welche Version von Windows Sie verwenden. (Öffnen Sie beispielsweise unter Windows 7 Systemsteuerung, System und Sicherheit, System (System)aus. Wählen Sie dann Erweiterte Systemeinstellungen, AdvancedRegisterkarte Umgebungsvariablen.) Weitere Informationen finden Sie in der Windows-Dokumentation.

Zusätzliche Ressourcen für den OIDC-Verband

Die folgenden Ressourcen können Ihnen helfen, mehr über den OIDC-Verbund zu erfahren:

- Verwenden Sie OpenID Connect in Ihren GitHub Workflows, indem Sie OpenID Connect in Amazon Web Services konfigurieren
- Amazon Cognito Identity im Amplify Libraries for Android-Handbuch und Amazon Cognito Identity im Amplify Libraries for Swift-Handbuch.
- Der Blog Automatisieren von OpenID Connect-basierten AWS IAM-Web-Identity-Rollen mit Microsoft Entra ID on the AWS Partner Network (APN) erklärt, wie automatisierte Hintergrundprozesse oder Anwendungen authentifiziert werden können, die außerhalb der OIDC-Autorisierung ausgeführt werden. AWS machine-to-machine

- Der Artikel Web Identity Federation with Mobile Applications behandelt den OIDC-Verbund und zeigt ein Beispiel für die Verwendung des OIDC-Verbunds, um Zugriff auf Inhalte in Amazon S3 zu erhalten.

SAML 2.0-Verbund

AWS unterstützt den Identitätsverbund mit SAML 2.0 (Security Assertion Markup Language 2.0), einem offenen Standard, den viele Identitätsanbieter (IdPs) verwenden. Diese Funktion ermöglicht föderiertes Single Sign-On (SSO), sodass sich Benutzer bei den API-Vorgängen anmelden AWS Management Console oder AWS API-Operationen aufrufen können, ohne dass Sie für jeden in Ihrer Organisation einen IAM-Benutzer erstellen müssen. Durch die Verwendung von SAML können Sie den Prozess der Konfiguration von Federation mit vereinfachen AWS, da Sie den IdP-Dienst verwenden können, anstatt benutzerdefinierten Identitätsproxycode zu schreiben.

Der IAM-Verbund unterstützt die folgenden Anwendungsfälle:

- Verbundzugriff, der es einem Benutzer oder einer Anwendung in Ihrer Organisation ermöglicht, API-Operationen aufzurufen AWS . Dieser Anwendungsfall wird im folgenden Abschnitt behandelt. Sie verwenden eine SAML-Zusicherung (als Teil der Authentifizierungsantwort), die in Ihrer Organisation generiert wird, um temporäre Sicherheitsanmeldeinformationen zu erhalten. Dieses Szenario ähnelt anderen Verbund-Szenarien, die von IAM unterstützt werden, wie diejenigen, die in Anfordern temporärer Sicherheitsanmeldeinformationen und OIDC-Föderation beschrieben werden. SAML 2.0, das IdPs in Ihrem Unternehmen basiert, verarbeitet jedoch viele Details zur Laufzeit für die Durchführung von Authentifizierungs- und Autorisierungsprüfungen.
- Webbasiertes Single Sign-On (SSO) für den AWS Management Console von Ihrem Unternehmen aus. Benutzer können sich bei einem Portal in Ihrer Organisation anmelden, das von einem SAML 2.0-kompatiblen IdP gehostet wird, eine Option auswählen, zu der sie wechseln möchten AWS, und zur Konsole weitergeleitet werden, ohne zusätzliche Anmeldeinformationen angeben zu müssen. Sie können außerdem einen SAML-Identitätsanbieter eines Drittanbieters verwenden, um SSO-Zugriff auf die Konsole zu erhalten. Oder Sie können einen benutzerdefinierten Identitätsanbieter erstellen, um externen Benutzern Zugriff auf die Konsole zu gewähren. Weitere Informationen zum Erstellen eines benutzerdefinierten Identitätsanbieters finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

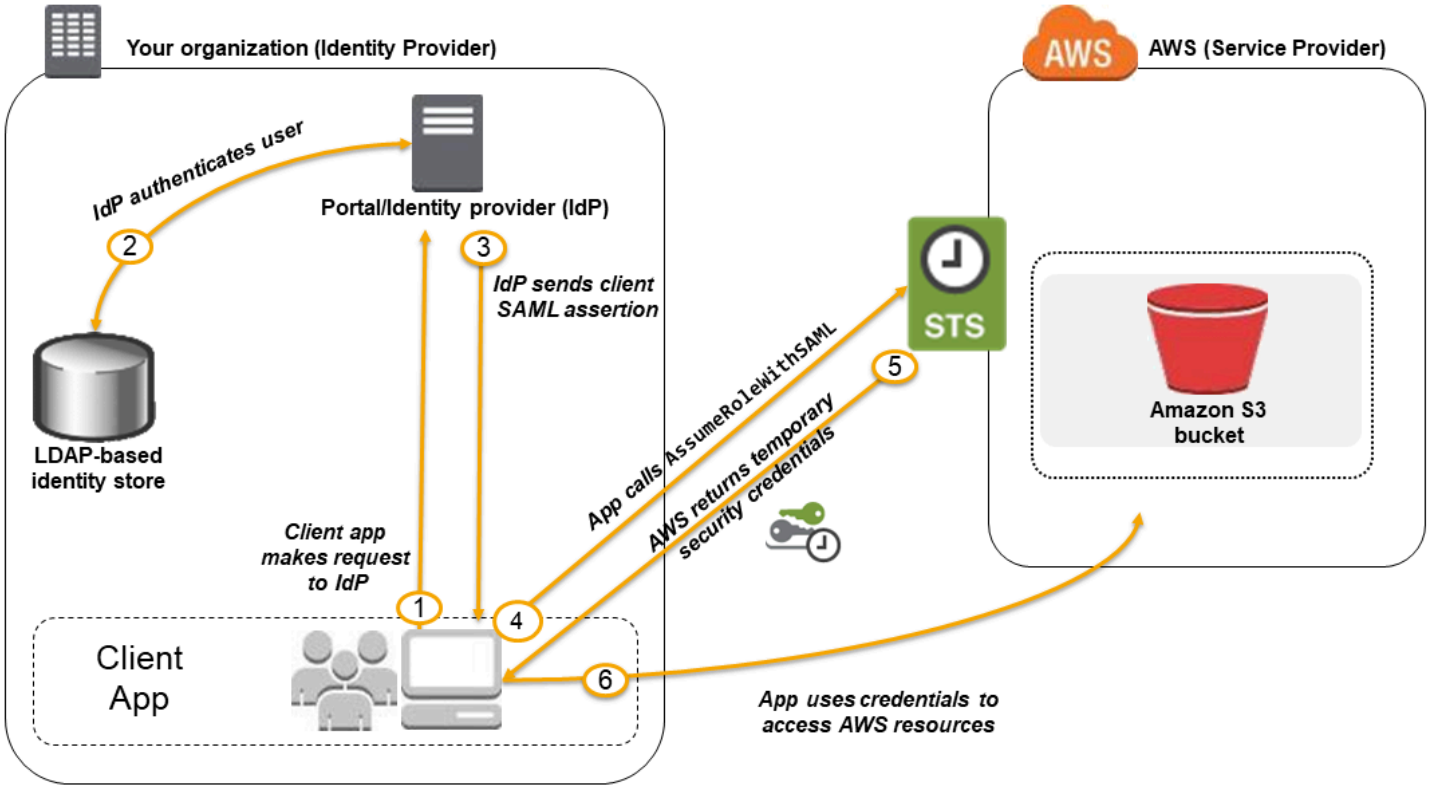
Themen

- Verwenden des SAML-basierten Verbunds für API-Zugriff auf AWS

- Überblick über die Konfiguration des SAML 2.0-basierten Verbunds
- Überblick über die Rolle, die den SAML-Verbundzugriff auf Ihre Ressourcen ermöglicht AWS
- Eindeutige Identifizierung von Benutzern im SAML-basierten Verbund
- Erstellen Sie einen SAML-Identitätsanbieter in IAM
- Konfigurieren Sie Ihren SAML 2.0-IdP mit dem Vertrauen der Vertrauensperson und dem Hinzufügen von Ansprüchen
- Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS
- SAML-Assertionen für die Authentifizierungsantwort konfigurieren
- Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console

Verwenden des SAML-basierten Verbunds für API-Zugriff auf AWS

Gehen wir davon aus, Sie möchten Mitarbeitern die Möglichkeit bieten, Daten von ihren Computern in einen Sicherungsordner zu kopieren. Sie erstellen eine Anwendung, die Benutzer auf ihrem Computer ausführen können. Am Backend liest und schreibt die Anwendung Objekte in einen Amazon S3 S3-Bucket. Benutzer haben keinen direkten Zugriff auf AWS. Stattdessen wird der folgende Prozess verwendet:



1. Ein Benutzer in Ihrer Organisation verwendet eine Client-App, um eine Authentifizierung vom Identitätsanbieter Ihrer Organisation anzufordern.
2. Der Identitätsanbieter authentifiziert den Benutzer anhand des Identitätsspeichers Ihrer Organisation.
3. Der Identitätsanbieter konstruiert eine SAML-Zusicherung mit Informationen über den Benutzer und sendet die Zusicherung an die Client-App.
4. Die Client-App ruft die AWS STS `AssumeRoleWithSAML`API auf und übergibt den ARN des SAML-Anbieters, den ARN der zu übernehmenden Rolle und die SAML-Assertion von IdP.
5. Die API-Antwort an die Client-App umfasst temporäre Sicherheitsanmeldeinformationen.
6. Die Client-App verwendet die temporären Sicherheitsanmeldeinformationen, um Amazon S3-API-Operationen aufzurufen.

Überblick über die Konfiguration des SAML 2.0-basierten Verbunds

Bevor Sie den SAML 2.0-basierten Verbund wie im vorherigen Szenario und Diagramm beschrieben verwenden können, müssen Sie den IdP Ihrer Organisation und Sie so konfigurieren, dass AWS-Konto sie sich gegenseitig vertrauen. Der allgemeine Prozess für die Konfiguration dieses Vertrauens wird in den folgenden Schritten beschrieben. Innerhalb Ihrer Organisation müssen Sie einen Identitätsanbieter haben, der SAML 2.0 unterstützt, wie Microsoft Active Directory Federation Service (AD FS, Teil von Windows Server), Shibboleth oder einen anderen kompatiblen SAML 2.0-Anbieter.

> **Note**
>
> Um die Verbundstabilität zu verbessern, empfehlen wir, dass Sie Ihren IdP und Ihren AWS -Verbund so konfigurieren, dass mehrere SAML-Anmeldeendpunkte unterstützt werden. Einzelheiten finden Sie im AWS Sicherheitsblogartikel How to use regional SAML endpoints for Failover.

Richten Sie den IdP Ihrer Organisation ein und vertrauen Sie AWS sich gegenseitig

1. Registrieren Sie sich AWS als Service Provider (SP) beim IdP Ihrer Organisation. Verwenden Sie das SAML-Metadatendokument von `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

   Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte).

Optional können Sie das SAML-Metadatendokument von `https://signin.aws.amazon.com/static/saml-metadata.xml` verwenden.

2. Mithilfe des Identitätsanbieters Ihrer Organisation erstellen Sie eine entsprechende Metadaten-XML-Datei, die Ihren Identitätsanbieter als IAM-Identitätsanbieter in AWS beschreibt. Es muss den Namen des Ausstellers, ein Erstellungs- und ein Ablaufdatum sowie Schlüssel enthalten, anhand derer die Authentifizierungsantworten (Assertionen) Ihrer Organisation validiert werden AWS können.
3. In der IAM-Konsole erstellen Sie einen SAML-Identitätsanbieter. Im Rahmen dieses Vorgangs laden Sie das SAML-Metadatendokument hoch, die vom IdP in Ihrer Organisation in erstellt wurden. Step 2 Weitere Informationen finden Sie unter Erstellen Sie einen SAML-Identitätsanbieter in IAM.
4. In IAM erstellen Sie eine oder mehrere IAM-Rollen. In der Vertrauensrichtlinie der Rolle legen Sie den SAML-Anbieter als Principal fest, wodurch eine Vertrauensbeziehung zwischen Ihrer Organisation und hergestellt wird. AWS Die Berechtigungsrichtlinie der Rolle legt fest, wozu die Benutzer in Ihrer Organisation in AWS berechtigt sind. Weitere Informationen finden Sie unter Erstellen von Rollen für externe Identitätsanbieter (Verbund).

   > **Note**
   >
   > SAML-IDPs, die in einer Rollen-Vertrauensrichtlinie verwendet werden, müssen sich in demselben Konto befinden, in dem sich die Rolle befindet.

5. Im Identitätsanbieter Ihrer Organisation definieren Sie Zusicherungen, in denen Benutzer oder Gruppen in Ihrer Organisation den IAM-Rollen zugeordnet werden. Beachten Sie, dass unterschiedliche Benutzer und Gruppen in Ihrer Organisation unterschiedlichen IAM-Rollen zugeordnet werden können. Die genauen Schritte zur Durchführung des Mappings hängen davon ab, welche Identitätsanbieter Sie verwenden. Im früheren Szenario eines Amazon S3-Ordners für Benutzer ist es möglich, dass alle Benutzer derselben Rolle zugeordnet werden, die Amazon S3-Berechtigungen bieten. Weitere Informationen finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren.

   Wenn Ihr IdP SSO für die AWS Konsole aktiviert, können Sie die maximale Dauer der Konsolensitzungen konfigurieren. Weitere Informationen finden Sie unter Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console.
6. In der Anwendung, die Sie erstellen, rufen Sie die AWS Security Token Service `AssumeRoleWithSAML` API auf und übergeben ihr den ARN des SAML-Anbieters, in dem Sie

erstellt habenStep 3, den ARN der Rolle, von der Sie annehmen sollen, dass Sie sie erstellt habenStep 4, und die SAML-Assertion über den aktuellen Benutzer, die Sie von Ihrem IdP erhalten. AWS stellt sicher, dass die Anfrage zur Übernahme der Rolle von dem IdP stammt, auf den im SAML-Anbieter verwiesen wird.

Weitere Informationen finden Sie unter AssumeRolewithSAML in der API-Referenz.AWS Security Token Service

7. Wenn die Anforderung erfolgreich ist, gibt die API eine Reihe temporärer Sicherheitsanmeldeinformationen zurück, die Ihre Anwendung verwenden kann, um signierte Anforderungen an AWS zu stellen. Ihre Anwendung hat Informationen über den aktuellen Benutzer und kann auf benutzerspezifische Ordner in Amazon S3 zugreifen, wie im vorherigen Szenario beschrieben.

Überblick über die Rolle, die den SAML-Verbundzugriff auf Ihre Ressourcen ermöglicht AWS

Die Rolle oder Rollen, die Sie in IAM erstellen, definieren, wozu Verbundbenutzer aus Ihrer Organisation in berechtigt sind AWS. Beim Erstellen der Vertrauensrichtlinie für die Rolle geben Sie den SAML-Anbieter an, die Sie zuvor als `Principal` erstellt haben. Darüber hinaus können Sie den Geltungsbereich der Vertrauensrichtlinie mithilfe einer `Condition` festlegen, sodass nur Benutzer, die bestimmte SAML-Attribute erfüllen, auf die Rolle zugreifen können. Sie können beispielsweise festlegen, dass nur Benutzer, deren SAML-Zugehörigkeit `staff` lautet (wie von https://openidp.feide.no bestätigt), Zugriff auf die Rolle haben, wie in der folgenden Musterrichtlinie gezeigt:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
```

```
  }]
}
```

> **Note**
>
> SAML-IDPs, die in einer Rollen-Vertrauensrichtlinie verwendet werden, müssen sich in demselben Konto befinden, in dem sich die Rolle befindet.

Weitere Informationen zu den SAML-Schlüsseln, die Sie in einer Richtlinie überprüfen können, finden Sie unter Verfügbare Schlüssel für SAML-basierten AWS STS -Verbund.

Sie können unter `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` regionale Endpunkte für das `saml:aud`-Attribut einfügen. Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte).

Für die Berechtigungsrichtlinie in der Rolle geben Sie Berechtigungen wie bei jeder anderen Rolle auch an. Wenn Benutzer aus Ihrer Organisation beispielsweise Amazon Elastic Compute Cloud-Instances verwalten dürfen, müssen Sie Amazon EC2 EC2-Aktionen in der Berechtigungsrichtlinie ausdrücklich zulassen, z. B. die Aktionen in der verwalteten FullAccessAmazonEC2-Richtlinie.

Eindeutige Identifizierung von Benutzern im SAML-basierten Verbund

Wenn Sie Zugriffsrichtlinien in IAM erstellen können, ist es häufig nützlich, wenn Berechtigungen basierend auf der Identität der Benutzer angegeben werden können. Für Benutzer, die mit SAML verbunden wurde, möchte eine Anwendung möglicherweise Informationen in Amazon S3 mit einer Struktur wie dieser behalten:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
```

Sie können den Bucket (`myBucket`) und den Ordner (`app1`) über die Amazon S3 S3-Konsole oder die erstellen AWS CLI, da es sich um statische Werte handelt. Die benutzerspezifischen Ordner (*`user1`*, *`user2`*, *`user3`* etc.) müssen jedoch mithilfe von Code zur Laufzeit erstellt werden, da der Wert, der den Benutzer identifiziert, erst bekannt wird, wenn sich der Benutzer das erste Mal über den Verbundprozess anmeldet.

Um Richtlinien zu schreiben, die auf benutzerspezifische Details als Teil eines Ressourcennamens verweisen, muss die Identität des Benutzers in SAML-Schlüsseln verfügbar sein, die in Richtlinienbedingungen verwendet werden können. Die folgenden Schlüssel sind für SAML 2.0-basierten Verbund zur Verwendung in IAM-Richtlinien verfügbar. Sie können die von den folgenden Schlüsseln zurückgegebenen Werte verwenden, um eindeutige Benutzerkennungen für Ressourcen wie Amazon S3-Ordner zu erstellen.

- `saml:namequalifier`. Ein Hash-Wert basierend auf der Verkettung des `Issuer`-Antwortwerts (`saml:iss`) und einer Zeichenfolge mit der AWS-Konto-ID und dem Anzeigenamen (der letzte Teil der ARN) des SAML-Anbieters in IAM. Die Verkettung der Konto-ID und des Anzeigenamens des SAML-Anbieters steht als der Schlüssel `saml:doc` für IAM-Richtlinien zur Verfügung. Die Konto-ID und der Name des Anbieters müssen durch ein '/' getrennt werden, wie in "123456789012/anbieter_name". Weitere Informationen finden Sie im `saml:doc`-Schlüssel unter Verfügbare Schlüssel für SAML-basierten AWS STS -Verbund.

  Die Kombination aus `NameQualifier` und `Subject` kann verwendet werden, um einen Verbundbenutzer eindeutig zu identifizieren. Die folgende Pseudocode zeigt, wie dieser Wert berechnet wird. In diesem Pseudocode bedeutet + eine Verkettung, `SHA1` stellt eine Funktion dar, die einen Nachrichtendigest mit SHA-1 produziert, und `Base64` stellt eine Funktion dar, die eine Base-64-codierte Version der Hash-Ausgabe produziert.

  ```
  Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
  MySAMLIdP" ) )
  ```

  Weitere Informationen über die Richtlinienschlüssel, die für den SAML-Verbund verfügbar sind, finden Sie unter Verfügbare Schlüssel für SAML-basierten AWS STS -Verbund.
- `saml:sub` (string). Dies ist der Betreff des Antrags, der einen Wert enthält, der einen einzelnen Benutzer innerhalb einer Organisation eindeutig identifiziert (zum Beispiel `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (string). Dieser Schlüssel kann `persistent`, `transient`oder die vollständige `Format`-URI aus den `Subject`- und `NameID`-Elementen sein, die in Ihrer SAML-Zusicherung verwendet wurden. Der Wert `persistent` gibt an, dass der Wert in `saml:sub` für einen Benutzer derselbe für alle Sessions ist. Wenn der Wert `transient` lautet, hat der Benutzer einen anderen `saml:sub`-Wert für jede Sitzung. Weitere Informationen zum `NameID`-Attribut des `Format`-Elements finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren.

Das folgende Beispiel zeigt eine Berechtigungsrichtlinie, die die oben genannten Schlüssel verwendet, um Berechtigungen für einen benutzerspezifischen Ordner in Amazon S3 zu gewähren. Die Richtlinie geht davon aus, dass die Amazon S3-Objekte mit einem Präfix identifiziert werden, das sowohl `saml:namequalifier` als auch `saml:sub` enthält. Beachten Sie, dass das `Condition`-Element einen Test umfasst, um sicherzustellen, dass `saml:sub_type` auf `persistent` festgelegt ist. Wenn er auf `transient` festgelegt ist, kann der `saml:sub`-Wert für den Benutzer für jede Sitzung unterschiedlich sein und die Kombination der Werte sollte nicht verwendet werden, um benutzerspezifische Ordner zu identifizieren.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}",
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

Weitere Informationen zum Mapping von Zusicherungen aus dem Identitätsanbieter zu Richtlinienschlüsseln finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren.

Erstellen Sie einen SAML-Identitätsanbieter in IAM

Ein IAM-SAML 2.0-Identitätsanbieter ist eine Entität in IAM, die einen externen Identitätsanbieter-Service (Identity Provider, IdP) beschreibt, der den Standard SAML 2.0 (Security Assertion Markup Language 2.0) unterstützt. Sie verwenden einen IAM-Identitätsanbieter, wenn Sie eine Vertrauensstellung zwischen einem SAML-kompatiblen IdP wie Shibboleth oder Active Directory Federation Services herstellen möchten AWS, damit Benutzer in Ihrer Organisation auf Ressourcen zugreifen können. AWS IAM-SAML-Identitätsanbieter werden als Auftraggeber in einer IAM-Vertrauensrichtlinie verwendet.

Weitere Informationen zu diesem Szenario finden Sie unter SAML 2.0-Verbund.

Sie können einen IAM-Identitätsanbieter in AWS Management Console oder mit AWS CLI Tools für Windows oder API-Aufrufen erstellen und verwalten. PowerShell AWS

Nach dem Erstellen eines SAML-Anbieters müssen Sie eine oder mehrere IAM-Rollen erstellen. Eine Rolle ist eine Identität AWS , die keine eigenen Anmeldeinformationen hat (wie dies bei einem Benutzer der Fall ist). Aber in diesem Kontext ist eine Rolle dynamisch einem Verbundbenutzer zugewiesen, der vom Identitätsanbieter der Organisation authentifiziert wird. Die Rolle ermöglicht es dem Identitätsanbieter Ihres Unternehmens, temporäre Sicherheitsanmeldeinformationen für den Zugriff auf AWS anzufordern. Die der Rolle zugewiesenen Richtlinien legen fest, was die Verbundbenutzer tun dürfen. AWS Weitere Informationen zum Erstellen einer Rolle für den SAML-Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter (Verbund).

Nachdem Sie die Rolle erstellt haben, schließen Sie die SAML-Vertrauensstellung ab, indem Sie Ihren IdP mit Informationen über AWS und die Rollen konfigurieren, die Ihre Verbundbenutzer verwenden sollen. Dies wird als Konfiguration der Vertrauensstellung zwischen Ihrem Identitätsanbieter und AWS bezeichnet. Weitere Informationen zum Konfigurieren der Vertrauensstellung für die vertrauende Seite finden Sie unter Konfigurieren Sie Ihren SAML 2.0-IdP mit dem Vertrauen der Vertrauensperson und dem Hinzufügen von Ansprüchen.

Themen

- Voraussetzungen
- Erstellen und verwalten Sie einen IAM-SAML-Identitätsanbieter (Konsole)
- Erstellen und verwalten Sie einen IAM-SAML-Identitätsanbieter ()AWS CLI
- Erstellen und verwalten Sie einen IAM-SAML-Identitätsanbieter (API)AWS

Voraussetzungen

Bevor Sie einen SAML-Identitätsanbieter erstellen können, benötigen Sie die folgenden Informationen von Ihrem IdP.

- Holen Sie sich das SAML-Metadatendokument von Ihrem IdP. Diese Metadatendatei enthält den Namen des Ausstellers, Ablaufinformationen und Schlüssel, die zum Überprüfen der vom IdP empfangenen SAML-Authentifizierung (Zusicherungen) verwendet werden können. Verwenden Sie zum Generieren des Metadatendokuments die von Ihrem externen IdP bereitgestellte Identitätsverwaltungssoftware.

> **Important**
>
> Diese Metadatendatei enthält den Namen des Ausstellers, Ablaufinformationen und Schlüssel, die zum Überprüfen der vom IdP empfangenen SAML-Authentifizierung (Zusicherungen) verwendet werden können. Die Metadatendatei muss im UTF-8-Format ohne BOM (Byte Order Mark, Markierung der Bytereihenfolge) codiert sein. Zum Entfernen der BOM können Sie die Datei mithilfe eines Textbearbeitung-Tools wie Notepad++ als UTF-8 codieren.
> Das x.509-Zertifikat, das Bestandteil des SAML-Metadatendokuments ist, muss eine Schlüsselgröße von mindestens 1024 Bit verwenden. Außerdem darf das x.509-Zertifikat auch keine wiederholten Erweiterungen enthalten. Sie können Erweiterungen verwenden, diese dürfen jedoch nur einmal im Zertifikat angezeigt werden. Wenn das x.509-Zertifikat keine der beiden Bedingungen erfüllt, schlägt die IdP-Erstellung fehl und gibt den Fehler „Unable to parse metadata“ (Metadaten können nicht analysiert werden) zurück.
> Gemäß der Definition im SAML V2.0 Metadata Interoperability Profile Version 1.0 bewertet IAM den Ablauf des X.509-Zertifikats des Metadatendokuments weder, noch ergreift es entsprechende Maßnahmen.

Anweisungen zur Konfiguration vieler verfügbarer IdPs Dateien, mit denen gearbeitet werden kann AWS, einschließlich der Generierung des erforderlichen SAML-Metadatendokuments, finden Sie unter. Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS

Hilfe zum SAML-Verbund finden Sie unter Problembehandlung beim SAML-Verbund.

Erstellen und verwalten Sie einen IAM-SAML-Identitätsanbieter (Konsole)

Sie können den verwenden, AWS Management Console um IAM-SAML-Identitätsanbieter zu erstellen, zu aktualisieren und zu löschen. Hilfe zum SAML-Verbund finden Sie unter Problembehandlung beim SAML-Verbund.

So erstellen Sie einen IAM-SAML-Identitätsanbieter (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. Wählen Sie im Navigationsbereich Identitätsanbieter und dann Anbieter erstellen.
3. Wählen Sie für die Anbieterkonfiguration SAML.

4. Geben Sie einen Namen für den Identitätsanbieter ein.
5. Klicken Sie unter Metadaten-Dokument auf Datei auswählen und geben Sie das SAML-Metadatendokument an, das Sie unter the section called “Voraussetzungen” heruntergeladen haben.
6. (Optional) Unter Tags hinzufügen können Sie Schlüssel-Wert-Paare hinzufügen, um Ihre zu identifizieren und zu organisieren. IdPs Sie können auch Tags verwenden, um den Zugriff auf AWS -Ressourcen zu steuern. Weitere Informationen zum Markieren von SAML-Identitätsanbietern finden Sie unter Markieren von IAM-SAML-Identitätsanbietern.

   Wählen Sie Add tag. Geben Sie Werte für jedes Tag-Schlüsselwertpaar ein.
7. Verifizieren Sie die angegebenen Informationen. Wenn Sie fertig sind, wählen Sie Anbieter hinzufügen.
8. Weisen Sie Ihrem Identitätsanbieter eine IAM-Rolle zu, um externen Benutzeridentitäten, die von Ihrem Identitätsanbieter verwaltet werden, Zugriff auf AWS Ressourcen in Ihrem Konto zu gewähren. Weitere Informationen zum Erstellen von Rollen für den Identitätsverbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter (Verbund).

   > **Note**
   >
   > SAML-IDPs, die in einer Rollen-Vertrauensrichtlinie verwendet werden, müssen sich in demselben Konto befinden, in dem sich die Rolle befindet.

So löschen Sie einen SAML-Anbieter (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. Wählen Sie im Navigationsbereich Identitätsanbieter.
3. Aktivieren Sie das Kontrollkästchen neben dem Identitätsanbieter, den Sie löschen möchten.
4. Wählen Sie Löschen aus. Ein neues Fenster wird geöffnet.
5. Bestätigen Sie, dass Sie den Anbieter löschen möchten, indem Sie das Wort `delete` in das Feld eingeben. Wählen Sie dann Löschen.

Erstellen und verwalten Sie einen IAM-SAML-Identitätsanbieter ()AWS CLI

Sie können den verwenden, AWS CLI um SAML-Anbieter zu erstellen, zu aktualisieren und zu löschen. Hilfe zum SAML-Verbund finden Sie unter Problembehandlung beim SAML-Verbund.

So erstellen Sie einen IAM-Identitätsanbieter und laden ein Metadatendokument hoch (AWS CLI)

- Führen Sie diesen Befehl aus: `aws iam create-saml-provider`

So aktualisieren Sie einen IAM-SAML-Identitätsanbieter ()AWS CLI

- Führen Sie diesen Befehl aus: `aws iam update-saml-provider`

Markieren eines vorhandenen IAM-Identitätsanbieters (AWS CLI)

- Führen Sie diesen Befehl aus: `aws iam tag-saml-provider`

Auflisten von Tags für bestehenden IAM-Identitätsanbieter (AWS CLI)

- Führen Sie diesen Befehl aus: `aws iam list-saml-provider-tags`

Entfernen von Tags auf einem bestehenden IAM-Identitätsanbieter (AWS CLI)

- Führen Sie diesen Befehl aus: `aws iam untag-saml-provider`

So löschen Sie einen IAM-SAML-Identitätsanbieter (AWS CLI)

1. (Optional) Führen Sie zum Auflisten der Informationen für alle Anbieter (z. B. ARN, Erstellungsdatum und Ablaufdatum) folgenden Befehl aus:
   - `aws iam list-saml-providers`
2. (Optional) Führen Sie den folgenden Befehl aus, um Informationen zu einem bestimmten Anbieter abzurufen, z. B. den ARN, das Erstellungsdatum, das Ablaufdatum, die Verschlüsselungseinstellungen und Informationen zum privaten Schlüssel:
   - `aws iam get-saml-provider`
3. Führen Sie zum Löschen eines IAM-Identitätsanbieters den folgenden Befehl aus:

- `aws iam delete-saml-provider`

Erstellen und verwalten Sie einen IAM-SAML-Identitätsanbieter (API)AWS

Sie können die AWS API verwenden, um SAML-Anbieter zu erstellen, zu aktualisieren und zu löschen. Hilfe zum SAML-Verbund finden Sie unter Problembehandlung beim SAML-Verbund.

So erstellen Sie einen IAM-Identitätsanbieter und laden ein Metadatendokument (API) hoch AWS

- Rufen Sie diese Operation auf: `CreateSAMLProvider`

Um einen IAM-SAML-Identitätsanbieter (API) zu aktualisieren AWS

- Rufen Sie diese Operation auf: `UpdateSAMLProvider`

So kennzeichnen Sie einen vorhandenen IAM-Identitätsanbieter (API)AWS

- Rufen Sie diese Operation auf: `TagSAMLProvider`

Um Tags für einen vorhandenen IAM-Identitätsanbieter (AWS API) aufzulisten

- Rufen Sie diese Operation auf: `ListSAMLProviderTags`

Um Tags auf einem vorhandenen IAM-Identitätsanbieter (AWS API) zu entfernen

- Rufen Sie diese Operation auf: `UntagSAMLProvider`

Um einen IAM-Identitätsanbieter (AWS API) zu löschen

1. (Optional) Rufen Sie den folgenden Vorgang auf IdPs, um Informationen für alle aufzulisten, z. B. den ARN, das Erstellungsdatum und das Ablaufdatum:

   - `ListSAMLProviders`
2. (Optional) Rufen Sie den folgenden Vorgang auf, um Informationen zu einem bestimmten Anbieter abzurufen, z. B. den ARN, das Erstellungsdatum, das Ablaufdatum, die Verschlüsselungseinstellungen und Informationen zum privaten Schlüssel:

- `GetSAMLProvider`

3. Rufen Sie zum Löschen eines Identitätsanbieters die folgende Operation auf:

   - `DeleteSAMLProvider`

Konfigurieren Sie Ihren SAML 2.0-IdP mit dem Vertrauen der Vertrauensperson und dem Hinzufügen von Ansprüchen

Wenn Sie einen IAM-Identitätsanbieter und die Rolle für den SAML-Zugriff erstellen, teilen Sie AWS mit, wer der externe Identitätsanbieter (Identity Provider, IdP) ist und was seine Benutzer tun dürfen. Ihr nächster Schritt besteht dann darin, dem IdP von seiner Rolle AWS als Dienstanbieter zu erzählen. Dieser Vorgang wird als Hinzufügen der Vertrauensstellung für die vertrauenden Seiten zwischen Ihrem Identitätsanbieter und AWS bezeichnet. Der genaue Prozess des Hinzufügens der Vertrauensstellung für die vertrauende Seite hängt davon ab, welchen Identitätsanbieter Sie verwenden. Weitere Informationen finden Sie in der Dokumentation Ihrer Identitätsverwaltungssoftware.

In vielen IdPs Fällen können Sie eine URL angeben, über die der IdP ein XML-Dokument lesen kann, das Informationen und Zertifikate der vertrauenden Partei enthält. Verwenden Sie für AWS`https://`*`region-code`*`.signin.aws.amazon.com/static/saml-metadata.xml` oder`https://signin.aws.amazon.com/static/saml-metadata.xml`. Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte).

Wenn Sie eine URL nicht direkt angeben können, laden Sie das XML-Dokument von der vorangegangenen URL herunter und importieren Sie es in Ihre Identitätsanbietersoftware.

Sie müssen auch entsprechende Anspruchsregeln in Ihrem IdP erstellen, die angeben AWS , dass es sich um eine vertrauende Partei handelt. Wenn der IdP eine SAML-Antwort an den AWS Endpunkt sendet, enthält diese eine SAML-Assertion, die einen oder mehrere Ansprüche enthält. Bei einem Anspruch handelt es sich um Informationen über den Benutzer und seine Gruppen. Eine Anspruchsregel ordnet diese Informationen in SAML-Attributen zu. Auf diese Weise können Sie sicherstellen, dass die SAML-Authentifizierungsantworten Ihres IdP die erforderlichen Attribute enthalten, die in IAM-Richtlinien zur Überprüfung der Berechtigungen für Verbundbenutzer AWS verwendet werden. Weitere Informationen finden Sie unter den folgenden Themen:

- Überblick über die Rolle, die den SAML-Verbundzugriff auf Ihre Ressourcen ermöglicht AWS. In diesem Thema erfahren Sie, wie SAML-spezifische Schlüssel in IAM-Richtlinien verwendet werden und wie sie genutzt werden, um Berechtigungen für SAML-Verbundbenutzer einzuschränken.
- SAML-Assertionen für die Authentifizierungsantwort konfigurieren. In diesem Thema wird erklärt, wie SAML-Ansprüche konfiguriert werden, die Informationen über den Benutzer enthalten. Die Ansprüche sind in einer SAML-Zusicherung gebündelt und in der SAML-Antwort, die an AWS gesendet wird, enthalten. Sie müssen sicherstellen, dass die AWS für Richtlinien erforderlichen Informationen in der SAML-Assertion in einer Form enthalten sind, die erkannt und verwendet werden kann. AWS
- Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS. Dieses Thema enthält Links zu Dokumentationen von Drittanbietern zur Integration von Identitätslösungen mit AWS.

> **Note**
>
> Um die Verbundstabilität zu verbessern, empfehlen wir, dass Sie Ihren IdP und Ihren AWS -Verbund so konfigurieren, dass mehrere SAML-Anmeldeendpunkte unterstützt werden. Einzelheiten finden Sie im AWS Sicherheitsblogartikel How to use regional SAML endpoints for Failover.

Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS

> **Note**
>
> Wir empfehlen, dass Sie von Ihren menschlichen Benutzern verlangen, beim Zugriff temporäre Anmeldeinformationen zu verwenden. AWS Haben Sie darüber nachgedacht, es zu verwenden AWS IAM Identity Center? Mit IAM Identity Center können Sie den Zugriff auf mehrere Konten zentral verwalten AWS-Konten und Benutzern von einem zentralen Ort aus MFA-geschützten Single Sign-On-Zugriff auf alle ihnen zugewiesenen Konten gewähren. Mit IAM Identity Center können Sie Benutzeridentitäten in IAM Identity Center erstellen und verwalten oder einfach eine Verbindung zu Ihrem vorhandenen SAML-2.0-kompatiblen Identitätsanbieter herstellen. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

Die folgenden Links helfen Ihnen, SAML 2.0-Identitätsanbieter (IdP) -Lösungen von Drittanbietern so zu konfigurieren, dass sie mit dem AWS Verbund funktionieren.

> **Tip**
>
> AWS Support-Techniker können Kunden mit Geschäfts- und Unternehmens-Supportplänen bei einigen Integrationsaufgaben unterstützen, die Software von Drittanbietern beinhalten. Eine aktuelle Liste der unterstützten Plattformen und Anwendungen erhalten Sie unter Welche Software anderer Anbieter wird unterstützt? auf der Seite AWS Support – Häufig gestellte Fragen.

Lösung	Weitere Informationen
Auth0	In Amazon Web Services integrieren — Diese Seite auf der Auth0-Dokumentationswebsite enthält Links zu Ressource n, die beschreiben, wie Single Sign-On (SSO) mit dem eingerichtet wird, AWS Management Console und enthält ein JavaScript Beispiel. Sie können Auth0 so konfigurieren, dass Sitzungs-Tags übergeben werden. Weitere Informati onen finden Sie unter Auth0 kündigt Partnerschaft mit vier AWS IAM-Sitzungs-Tags an.
Microsoft Entra	Tutorial: Microsoft Entra SSO-Integration mit AWS Einzelkontenzugriff — In diesem Tutorial auf der Microsoft -Website wird beschrieben, wie Microsoft Entra (früher bekannt als Azure AD) mithilfe des SAML-Verbunds als Identitätsanbieter (IdP) eingerichtet wird.
Centrify	Centrify konfigurieren und SAML für SSO verwenden AWS — Auf dieser Seite der Centrify-Website wird erklärt, wie Centrify so konfiguriert wird, dass SAML für SSO verwendet wird. AWS
CyberArk	Konfigurieren Sie CyberArk, um Benutzern, die sich über SAML Single Sign-On (SSO AWS) vom CyberArk

Lösung	Weitere Informationen
	Benutzerportal aus anmelden, Zugriff auf Amazon Web Services () zu gewähren.
ForgeRock	Die ForgeRock Identity Platform lässt sich integrieren in. AWS Sie können so konfigurieren ForgeRock , dass Sitzungs-Tags übergeben werden. Weitere Informati onen finden Sie unter Atributbasierte Zugriffskontrolle für Amazon Web Services.
Google WorkSpace	Cloud-Anwendung Amazon Web Services — In diesem Artikel auf der Google Workspace-Admin-Hilfeseite wird beschrieben, wie Sie Google Workspace als SAML 2.0-IdP mit Service AWS Provider konfigurieren.
IBM	Sie können IBM so konfigurieren, dass Sitzungs-Tags übergeben werden. Weitere Informationen finden Sie unter IBM Cloud Identity IDaaS, eines der ersten Unternehmen, das Sitzungs-Tags unterstützt. AWS
JumpCloud	Zugriff über IAM-Rollen für Single Sign On (SSO) mit Amazon gewähren AWS — In diesem Artikel auf der JumpCloud Website wird beschrieben, wie SSO auf der Grundlage von IAM-Rollen für eingerichtet und aktiviert wird. AWS
Matrix42	MyWorkspace Leitfaden für die ersten Schritte — In diesem Handbuch wird beschrieben, wie AWS Identitätsdienste in Matrix42 integriert werden. MyWorkspace

Lösung	Weitere Informationen
Microsoft Active Directory Federation Services (AD FS)	Praktische Hinweise: Integration von Active Directory Federation Service mit AWS IAM Identity Center — In diesem Beitrag im AWS Architektur-Blog wird der Authentifizierungsablauf zwischen AD FS und AWS IAM Identity Center (IAM Identity Center) erläutert. IAM Identity Center unterstützt den Identitätsverbund mit SAML 2.0 und ermöglicht die Integration mit AD FS-Lösungen. Benutzer können sich mit ihren Unternehmens-Anmeldeinformationen im IAM Identity Center-Portal anmelden, wodurch der Aufwand für die Verwaltung separater Anmeldeinformationen auf das IAM Identity Center reduziert wird. Sie können AD FS auch so konfigurieren, dass es Sitzungs-Tags weiterleitet. Weitere Informationen finden Sie unter Verwenden der attributbasierten Zugriffssteuerung mit AD FS, um die IAM-Berechtigungsverwaltung zu vereinfachen.
miniOrange	SSO für AWS — Auf dieser Seite der MiniOrange-Website wird beschrieben, wie ein sicherer Zugriff AWS für Unternehmen und die vollständige Kontrolle über den Zugriff auf AWS Anwendungen eingerichtet werden können.
Okta	Integration der Amazon Web Services Command Line Interface mit Okta – Auf dieser Seite der Okta-Support-Website erfahren Sie, wie Sie Okta für die Verwendung mit AWS. Sie können Okta so konfigurieren, dass Sitzungs-Tags übergeben werden. Weitere Informationen finden Sie unter Okta und AWS Partner to Simplify Access via Session Tags.
Okta	AWS Kontoverbund — In diesem Abschnitt auf der Okta-Website wird beschrieben, wie Sie IAM Identity Center für einrichten und aktivieren. AWS

Lösung	Weitere Informationen
OneLogin	Suchen Sie in der OneLoginKnowledgebase nach **SAML AWS** einer Liste von Artikeln, in denen erklärt wird, wie Sie die IAM Identity Center-Funktionalität zwischen OneLogin und AWS für Einzelrollen- und Mehrrollenszenarien einrichten. Sie können so konfigurieren, dass Sitzungs-Tags übergeben werden. OneLogin Weitere Informationen finden Sie unter OneLogin und Sitzungs-Tags: Attributb asierte Zugriffskontrolle für Ressourcen. AWS
Ping Identity	PingFederate AWS Connector — Hier finden Sie Details zum PingFederate AWS Connector, einer Schnellve rbindungsvorlage, mit der Sie auf einfache Weise Single Sign-On (SSO) und Bereitstellungsverbindung einrichten können. Lesen Sie die Dokumentation und laden Sie den neuesten PingFederate AWS Connector für Integrationen mit herunter. AWS Sie können Ping Identity so konfiguri eren, dass Sitzungs-Tags übergeben werden. Weitere Informationen finden Sie unter Ankündigung der Ping-Iden titätsunterstützung für die attributbasierte Zugriffskontrolle in AWS.
RadiantLogic	Technologiepartner von Radiant Logic — Der RadiantOn e Federated Identity Service von Radiant Logic lässt sich integrieren und bietet so einen Identitätsknotenpunkt für SAML-basiertes SSO. AWS
RSA	Der Amazon Web Services — RSA Ready Implementation Guide bietet Anleitungen zur Integration AWS und RSA. Weitere Informationen zur SAML-Konfiguration finden Sie unter Amazon Web Services — SAML My Page SSO Configuration — RSA Ready Implementation Guide.

Lösung	Weitere Informationen
Salesforce.com	So konfigurieren Sie SSO von Salesforce bis AWS — In diesem How-to-Artikel auf der Entwicklerseite von Salesforce.com wird beschrieben, wie Sie einen Identität sanbieter (IdP) in Salesforce einrichten und als Dienstanb ieter konfigurieren AWS .
SecureAuth	AWS - SecureAuth SAML SSO — In diesem Artikel auf der SecureAuth Website wird beschrieben, wie die SAML-Integration für eine Appliance eingerichtet wird. AWS SecureAuth
Shibboleth	So verwenden Sie Shibboleth für SSO zum AWS Management Console— Dieser Eintrag im AWS Sicherhei ts-Blog enthält ein step-by-step Tutorial zur Einrichtung von Shibboleth und zur Konfiguration als Identitätsanbieter für. AWS Sie können Shibboleth so konfigurieren, dass Sitzungs-Tags übergeben werden.

Weitere Informationen finden Sie auf der Seite mit den IAM-Partnern auf der Website. AWS

SAML-Assertionen für die Authentifizierungsantwort konfigurieren

Nachdem Sie die Identität eines Benutzers in Ihrer Organisation überprüft haben, sendet der externe Identitätsanbieter (IdP) eine Authentifizierungsantwort an den AWS SAML-Endpunkt unter. `https://region-code.signin.aws.amazon.com/saml` Eine Liste potentieller Stellvertreter für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte). Diese Antwort ist eine POST-Anforderung, die ein SAML-Token enthält, das dem Standard für HTTP-POST-Bindungen für SAML 2.0 entspricht und die folgenden Elemente oder Ansprüche enthält. Sie konfigurieren diese Ansprüche in Ihrem SAML-kompatiblen Identitätsanbieter. Anweisungen dazu, wie Sie diese Ansprüche eingeben, finden Sie in der Dokumentation für Ihren Identitätsanbieter.

Wenn der IdP die Antwort mit den Ansprüchen an sendet AWS, werden viele der eingehenden Ansprüche AWS Kontextschlüsseln zugeordnet. Diese Kontextschlüssel können in IAM-Richtlinien mithilfe des `Condition`-Elements überprüft werden. Eine Liste der verfügbaren Zuordnungen folgt im Abschnitt Zuordnung von SAML-Attributen zu AWS vertrauenswürdigen Policy-Kontextschlüsseln.

`Subject` und `NameID`

Der folgende Auszug zeigt ein Beispiel. Ersetzen Sie die markierten Werte durch Ihre eigenen Werte. Es muss genau ein `SubjectConfirmation`-Element mit einem `SubjectConfirmationData`-Element geben, das das `NotOnOrAfter`-Attribut und das `Recipient`-Attribut enthält.
Zu diesen Attributen gehört ein Wert, der mit dem AWS Endpunkt `https://`*`region-code`*`.signin.aws.amazon.com/saml` übereinstimmen muss. Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte). Für den AWS Wert können Sie auch verwendenhttps://signin.aws.amazon.com/static/saml, wie im folgenden Beispiel gezeigt.

`NameID`-Elemente können den Wert „persistent“ oder „transient“ haben oder aus dem vollständigen Format-URI bestehen, der von der IDP-Lösung bereitgestellt wird. Der Wert „persistent“ gibt an, dass der Wert in `NameID` für einen Benutzer für alle Sitzungen identisch ist. Wenn der Wert „transient“ lautet, hat der Benutzer für jede Sitzung einen anderen `NameID`-Wert. Single-Sign-On-Interaktionen unterstützen die folgenden Arten von Identifikatoren:

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://signin.aws.amazon.com/saml"/>
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> Der `saml:aud`-Kontextschlüssel stammt aus dem Empfänger-Attribut in SAML, da dies das SAML-Äquivalent zum OIDC-Zielgruppenfeld ist, z. B. `accounts.google.com:aud`.

`PrincipalTag`-SAML-Attribut

(Optional) Sie können ein `Attribute`-Element verwenden, bei dem das `Name`-Attribut auf `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`* festgelegt ist. Mit diesem Element können Sie Attribute als Sitzungs-Tags in der SAML-Zusicherung übergeben. Weitere Hinweise zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

Um Attribute als Sitzungs-Tags zu übergeben, schließen Sie das `AttributeValue`-Element ein, das den Wert des Tags angibt. Verwenden Sie zum Beispiel das folgende Attribut, um die Tag-Schlüssel-Wert-Paare `Project` = `Marketing` und `CostCenter` = `12345` zu übergeben. Fügen Sie für jedes Tag ein separates `Attribute`-Element hinzu.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Um die oben genannten Tags als transitiv festzulegen, schließen Sie ein weiteres `Attribute`-Element ein und legen dabei für das Attribut `Name` den Wert`https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys` fest. Dies ist ein optionales Mehrwertattribut, das Ihre Sitzungs-Tags als transitiv festlegt. Transitive Tags bleiben erhalten, wenn Sie die SAML-Sitzung verwenden, um eine andere Rolle in AWS zu übernehmen. Dies wird als Rollenverkettung bezeichnet. Um beispielsweise die Tags `Principal` und `CostCenter` als transitiv festzulegen, verwenden Sie das folgende Attribut, um die Schlüssel anzugeben.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

`Role`-SAML-Attribut

Sie können ein `Attribute`-Element verwenden, bei dem das `Name`-Attribut auf `https://aws.amazon.com/SAML/Attributes/Role` festgelegt ist. Dieses Element enthält mindestens ein `AttributeValue`-Element, das den IAM-Identitätsanbieter und die Rolle aufführt, denen der Benutzer von Ihrem Identitätsanbieter zugeordnet wird. Die IAM-Rolle und der IAM-Identitätsanbieter werden als kommagetrenntes Paar von ARNs im gleichen Format wie die Parameter `RoleArn` und `PrincipalArn` angegeben, die an SAML übergeben werden. AssumeRoleWith Dieses Element muss mindestens ein Rolle-Anbieter-Paar (`AttributeValue`-Element) enthalten und kann auch mehrere Paare enthalten. Wenn das Element mehrere Paare enthält, werden die Benutzer aufgefordert auszuwählen, welche Rolle übernommen werden soll, wenn sie sich mithilfe von WebSSO bei der AWS Management Console anmelden.

> **Important**
>
> Beim Wert des `Name`-Attributs im `Attribute`-Tag ist die Groß-/Kleinschreibung zu beachten. Er muss ganz genau auf `https://aws.amazon.com/SAML/Attributes/Role` festgelegt werden.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

`RoleSessionName`-SAML-Attribut

Sie können ein `Attribute`-Element verwenden, bei dem das `Name`-Attribut auf `https://aws.amazon.com/SAML/Attributes/RoleSessionName` festgelegt ist. Dieses Element enthält ein `AttributeValue`-Element, das einen Bezeichner für die temporären -Anmeldeinformationen bereitstellt, die für SSO ausgestellt werden. Damit können Sie die temporären Anmeldeinformationen dem Benutzer zuordnen, der Ihre Anwendung verwendet. Dieses Element wird verwendet, um Benutzerinformationen in der anzuzeigen. AWS Management Console Der Wert des `AttributeValue`-Elements muss zwischen 2 und 64 Zeichen lang sein und darf nur

alphanumerische Zeichen, Unterstriche und die folgenden Zeichen enthalten: . , + = @ - (Bindestrich). Er darf keine Leerzeichen enthalten. Der Wert ist in der Regel eine Benutzer-ID (`johndoe`) oder eine E-Mail-Adresse (`johndoe@example.com`). Er sollte kein Wert sein, der ein Leerzeichen enthält, wie etwa der Anzeigename eines Benutzers (`John Doe`).

> **Important**
>
> Beim Wert des `Name`-Attributs im `Attribute`-Tag ist die Groß-/Kleinschreibung zu beachten. Er muss ganz genau auf `https://aws.amazon.com/SAML/Attributes/RoleSessionName` festgelegt werden.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

`SessionDuration`-SAML-Attribut

(Optional) Sie können ein `Attribute`-Element verwenden, bei dem das `Name`-Attribut auf `https://aws.amazon.com/SAML/Attributes/SessionDuration"` festgelegt ist. Dieses Element enthält ein `AttributeValue` Element, das angibt, wie lange der Benutzer darauf zugreifen kann, AWS Management Console bevor er neue temporäre Anmeldeinformationen anfordern muss. Der Wert ist eine Ganzzahl, die die Anzahl der Sekunden für die Sitzung angibt. Der Wert kann im Bereich zwischen 900 Sekunden (15 Minuten) und 43 200 Sekunden (12 Stunden) liegen. Wenn dieses Attribut nicht vorhanden ist, gelten die Anmeldeinformationen eine Stunde lang (der Standardwert des `DurationSeconds`-Parameters der `AssumeRoleWithSAML`-API).

Um dieses Attribut verwenden zu können, müssen Sie den SAML-Anbieter so konfigurieren, dass er Single Sign-On-Zugriff auf den Web-Endpunkt für die Anmeldung AWS Management Console über die Konsole unter ermöglicht. `https://region-code.signin.aws.amazon.com/saml` Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte). Optional können Sie die folgende URL verwenden: `https://signin.aws.amazon.com/static/saml`. Beachten Sie, dass dieses Attribut nur Sitzungen in der AWS Management Console verlängert. Es kann nicht die Lebensdauer anderer Anmeldeinformationen verlängern. Wenn es jedoch in einem `AssumeRoleWithSAML`-API-Aufruf vorhanden ist, kann es verwendet werden, um die Dauer der Sitzung zu verkürzen. Die standardmäßige Gültigkeitsdauer der vom Aufruf zurückgegebenen Anmeldeinformationen beträgt 60 Minuten.

Beachten Sie außerdem, dass wenn auch ein `SessionNotOnOrAfter`-Attribut definiert ist, der niedrigere Wert der beiden Attribute, `SessionDuration` oder `SessionNotOnOrAfter`, die maximale Dauer der Konsolensitzung bestimmt.

Wenn Sie Konsolensitzungen mit einer erweiterten Dauer aktivieren, steigt das Risiko, dass die Sicherheit der Anmeldeinformationen gefährdet wird. Um dieses Risiko zu minimieren, können Sie die aktiven Konsolensitzungen für jede Rolle direkt deaktivieren, indem Sie Revoke Sessions auf der Seite Role Summary in der IAM-Konsole wählen. Weitere Informationen finden Sie unter Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen.

> **Important**
>
> Beim Wert des `Name`-Attributs im `Attribute`-Tag ist die Groß-/Kleinschreibung zu beachten. Er muss ganz genau auf `https://aws.amazon.com/SAML/Attributes/SessionDuration` festgelegt werden.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

`SourceIdentity`-SAML-Attribut

(Optional) Sie können ein `Attribute`-Element verwenden, bei dem das `Name`-Attribut auf `https://aws.amazon.com/SAML/Attributes/SourceIdentity` festgelegt ist. Dieses Element enthält ein `AttributeValue`-Element, das einen Identifikator für die Person oder Anwendung bereitstellt, die eine IAM-Rolle verwendet. Der Wert für die Quellidentität bleibt bestehen, wenn Sie die SAML-Sitzung verwenden, um eine andere Rolle zu übernehmen, was als Rollenverkettung bezeichnet wird. AWS Der Wert für die Quellidentität ist in der Anforderung für jede Aktion vorhanden, die während der Rollensitzung ausgeführt wird. Der festgelegte Wert kann während der Rollensitzung nicht geändert werden. Administratoren können dann AWS CloudTrail Protokolle verwenden, um die Quellidentitätsinformationen zu überwachen und zu prüfen, um festzustellen, wer Aktionen mit gemeinsam genutzten Rollen ausgeführt hat.

Der Wert des `AttributeValue`-Elements muss zwischen 2 und 64 Zeichen lang sein und darf nur alphanumerische Zeichen, Unterstriche und die folgenden Zeichen enthalten: . , + = @ - (Bindestrich). Er darf keine Leerzeichen enthalten. Der Wert ist normalerweise ein Attribut, das dem Benutzer zugeordnet ist, z. B. eine Benutzer-ID (`johndoe`) oder eine E-Mail-Adresse

(`johndoe@example.com`) enthalten. Er sollte kein Wert sein, der ein Leerzeichen enthält, wie etwa der Anzeigename eines Benutzers (`John Doe`). Weitere Informationen zur Verwendung von identitätsbasierten Richtlinien finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

> **Important**
>
> Wenn Ihre SAML-Zusicherung für die Verwendung des `SourceIdentity`-Attributs konfiguriert ist, muss Ihre Vertrauensrichtlinie auch die `sts:SetSourceIdentity`-Aktion enthalten. Weitere Informationen zur Verwendung von identitätsbasierten Richtlinien finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

Um ein Quellidentitätsattribut zu übergeben, schließen Sie das `AttributeValue`-Element ein, das den Wert der Quellidentität angibt. Um zum Beispiel die Quellidentität `DiegoRamirez` zu übergeben, verwenden Sie das folgende Attribut.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>DiegoRamirez</AttributeValue>
```

Zuordnung von SAML-Attributen zu AWS vertrauenswürdigen Policy-Kontextschlüsseln

Die Tabellen in diesem Abschnitt enthalten häufig genutzte SAML-Attribute und deren Zuordnung zu Bedingungskontextschlüsseln von Vertrauensrichtlinien in AWS. Mit diesen Schlüsseln können Sie den Zugriff auf eine Rolle steuern. Vergleichen Sie dazu die Schlüssel mit den Werten, die in den eine SAML-Zugriffsanforderung begleitenden Zusicherungen enthalten sind.

> **Important**
>
> Diese Schlüssel sind nur in IAM-Vertrauensrichtlinien verfügbar (Richtlinien, die bestimmen, wer eine Rolle übernehmen kann) und gelten nicht für Berechtigungsrichtlinien.

In der Tabelle mit den eduPerson- und eduOrg-Attributen werden Werte entweder als Zeichenfolgen oder als Listen von Zeichenfolgen eingegeben. Bei Zeichenfolgenwerten können Sie diese Werte in IAM-Vertrauensrichtlinien mithilfe der `StringEquals`- oder der `StringLike`-Bedingung testen. Bei Werten, die eine Liste von Zeichenfolgen enthalten, können Sie die `ForAnyValue`- und

`ForAllValues` -Richtlinienmengenoperatoren verwenden, um die Werte in Vertrauensrichtlinien zu testen.

> **Note**
>
> Sie sollten nur einen Anspruch pro AWS Kontextschlüssel angeben. Wenn Sie mehr als einen Anspruch einfügen, wird nur ein einziger Anspruch zugeordnet.

eduPerson- und eduOrg-Attribute

eduPerson- oder eduOrg-Attribut (**`Name`**-Schlüssel)	Ordnet diesem AWS Kontextschlüssel (**`FriendlyName`** Schlüssel) zu	Typ
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPersonAffiliation`	Liste von Zeichenfolgen
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Liste von Zeichenfolgen
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPersonOrgUnitDN`	Liste von Zeichenfolgen
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPersonPrimaryAffiliation`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPersonPrincipalName`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPersonEntitlement`	Liste von Zeichenfolgen
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPersonPrimaryOrgUnitDN`	String

eduPerson- oder eduOrg-Attribut (**Name**-Schlüssel)	Ordnet diesem AWS Kontextschlüssel (**FriendlyName** Schlüssel) zu	Typ
urn:oid:1.3.6.1.4.1.5923.1.1.1.9	eduPersonScopedAffiliation	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.1.1.10	eduPersonTargetedID	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.1.1.11	eduPersonAssurance	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.2.1.2	eduOrgHomePageURI	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.2.1.3	eduOrgIdentityAuthNPolicyURI	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.2.1.4	eduOrgLegalName	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.2.1.5	eduOrgSuperiorURI	Liste von Zeichenfolgen
urn:oid:1.3.6.1.4.1.5923.1.2.1.6	eduOrgWhitePagesURI	Liste von Zeichenfolgen
urn:oid:2.5.4.3	cn	Liste von Zeichenfolgen

Active Directory-Attribute

AD-Attribut	Ordnet diesem AWS Kontextschlüssel zu	Typ
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name`	`name`	Zeichenfolge
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	Zeichenfolge
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	Zeichenfolge
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	Zeichenfolge
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	Zeichenfolge
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	String

X.500-Attribute

X.500-Attribut	Ordnet diesem AWS Kontextschlüssel zu	Typ
`2.5.4.3`	`commonName`	Zeichenfolge
`2.5.4.4`	`surname`	Zeichenfolge
`2.4.5.42`	`givenName`	Zeichenfolge
`2.5.4.45`	`x500UniqueIdentifier`	Zeichenfolge
`0.9.2342.19200300100.1.1`	`uid`	Zeichenfolge

X.500-Attribut	Ordnet diesem AWS Kontextschlüssel zu	Typ
`0.9.2342.19200300100.1.3`	`mail`	Zeichenfolge
`0.9.2342.19200300.100.1.45`	`organizationStatus`	String

Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console

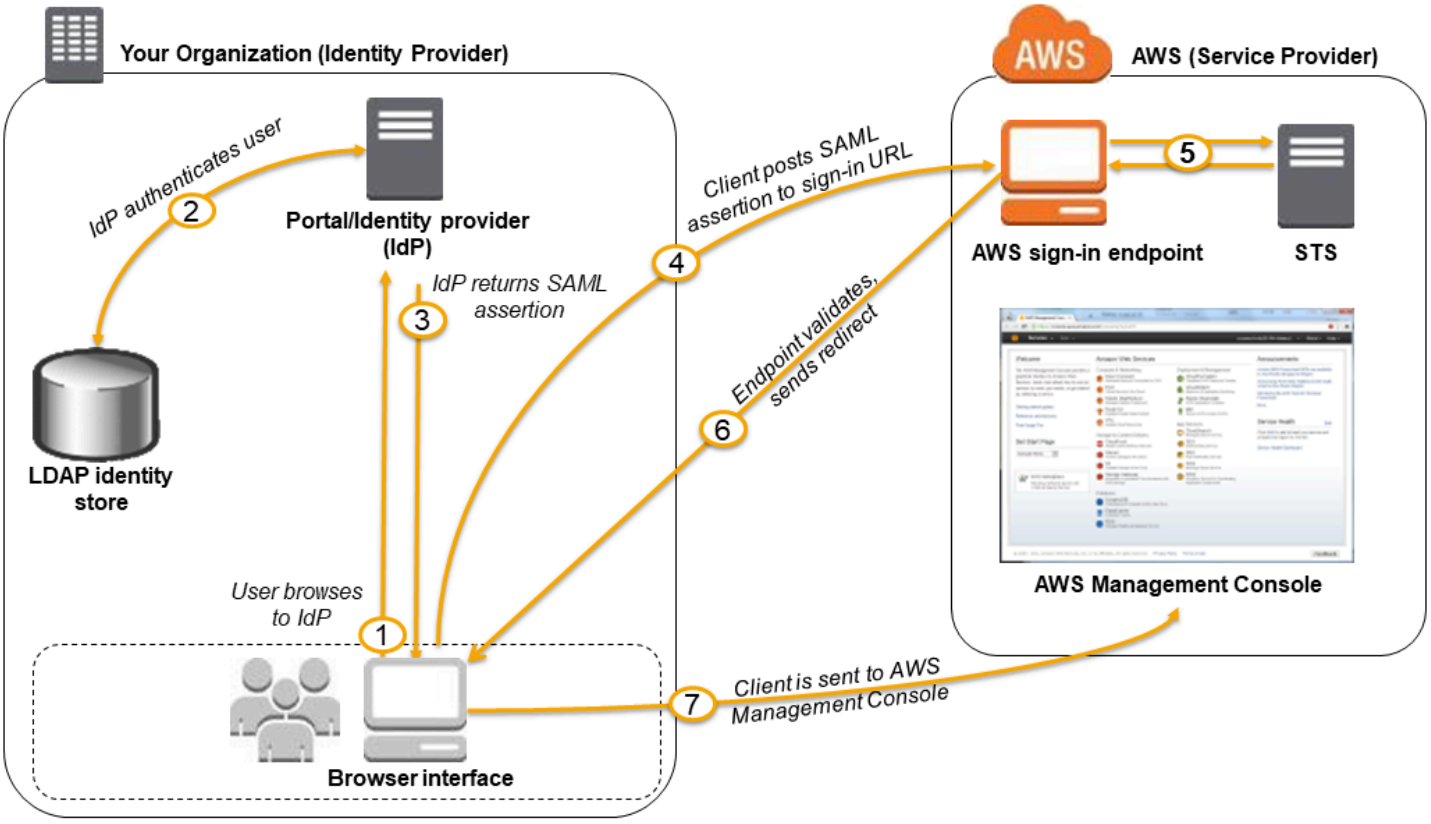
Sie können eine Rolle verwenden, um Ihren SAML 2.0-kompatiblen Identitätsanbieter (IdP) zu konfigurieren und Ihren Verbundbenutzern den Zugriff AWS auf zu ermöglichen. AWS Management Console Die Rolle gewährt dem Benutzer die Berechtigung, bestimmte Aufgaben in der Konsole auszuführen. Wenn Sie verbundenen SAML-Benutzern anderweitig Zugriff auf AWS gewähren möchten, beachten Sie die folgenden Themen:

- AWS CLI: Wechseln zu einer IAM-Rolle (AWS CLI)
- Tools für Windows: PowerShell Zu einer IAM-Rolle wechseln (Tools für Windows PowerShell)
- AWS API: Zu einer IAM-Rolle (AWS API) wechseln

Übersicht

Die folgende Abbildung zeigt den Ablauf von SAML-fähigem Single-Sign-On-Zugriff.

> **Note**
>
> Diese spezielle Verwendung von SAML unterscheidet sich von der allgemeineren Verwendung, die unter dargestellt istSAML 2.0-Verbund, da dieser Workflow das AWS Management Console im Namen des Benutzers öffnet. Hierfür müssen Sie den AWS -Anmelde-Endpunkt verwenden, statt die `AssumeRoleWithSAML`-API direkt aufzurufen. Der Endpunkt ruft die API für den Benutzer auf und gibt eine URL zurück, über die der Browser des Benutzers automatisch an die AWS Management Console geleitet wird.



Die Abbildung zeigt die folgenden Schritte:

1. Der Benutzer öffnet das Portal Ihrer Organisation und wählt die Option aus, zur AWS Management Console zu wechseln. In Ihrer Organisation ist das Portal in der Regel eine Funktion Ihres IdP, der den Vertrauensaustausch zwischen Ihrer Organisation und AWS abwickelt. Die Portal-URL für Active Directory Federation Services lautet beispielsweise: `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`.
2. Das Portal überprüft die Identität des Benutzers in Ihrer Organisation.
3. Das Portal generiert eine SAML-Authentifizierungsantwort, die eine Zusicherung enthält, über die der Benutzer identifiziert wird und die Attribute zu dem Benutzer enthält. Sie können in der Konfiguration Ihres Identitätsanbieters das SAML-Zusicherungsattribut `SessionDuration` aufnehmen, um die maximale Länge der Konsolensitzung festzulegen. Sie können den Identitätsanbieter auch so konfigurieren, dass Attribute als Sitzungs-Tags übergeben werden. Das Portal sendet diese Antwort an den Client-Browser.
4. Der Client-Browser wird zum AWS Single Sign-On-Endpunkt umgeleitet und veröffentlicht die SAML-Assertion.
5. Der Endpunkt fordert temporäre Sicherheitsanmeldeinformationen für den Benutzer an und erstellt eine Konsolenanmelde-URL, in der diese Anmeldeinformationen verwendet werden.

6. AWS sendet die Anmelde-URL als Umleitung zurück an den Client.
7. Der Client-Browser wird an die AWS Management Console weitergeleitet. Wenn die SAML-Authentifizierungsantwort Attribute enthält, die mehrere IAM-Rollen abbilden, wird der Benutzer zunächst aufgefordert, die Rolle für den Zugriff auf die Konsole auszuwählen.

Aus Sicht des Benutzers läuft der Prozess transparent ab: Der Benutzer beginnt im internen Portal Ihrer Organisation und endet am Ende im AWS Management Console, ohne jemals Anmeldeinformationen angeben zu müssen. AWS

Die folgenden Abschnitte enthalten einen Überblick über die Konfiguration dieses Verhaltens sowie Links zu detaillierten Anleitungen.

Konfigurieren Sie Ihr Netzwerk als SAML-Anbieter für AWS

Konfigurieren Sie im Netzwerk Ihrer Organisation den Identitätsspeicher (z. B. Windows Active Directory) für die Zusammenarbeit mit einem SAML-basierten Identitätsanbieter, wie etwa Windows Active Directory Federation Services, Shibboleth usw. Mithilfe Ihres Identitätsanbieters generieren Sie ein Metadatendokument, in dem Ihre Organisation als Identitätsanbieter beschrieben wird und das Authentifizierungsschlüssel enthält. Sie konfigurieren das Portal Ihrer Organisation auch so, dass Benutzeranfragen für die Authentifizierung mithilfe von SAML-Assertionen AWS Management Console an den AWS SAML-Endpunkt weitergeleitet werden. Die konkrete Konfiguration des Identitätsanbieters zur Bereitstellung der Datei metadata.xml ist abhängig von Ihrem Identitätsanbieter. Eine Anleitung finden Sie in der Dokumentation Ihres Identitätsanbieters. Unter Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS finden Sie Links zur Onlinedokumentation für viele unterstützte SAML-Anbieter.

Erstellen eines SAML-Anbieters in IAM

Als Nächstes melden Sie sich bei der an AWS Management Console und gehen zur IAM-Konsole. Erstellen Sie dort einen neuen SAML-Anbieter. Dabei handelt es sich um eine Entität in IAM, in der die Informationen zum Identitätsanbieter Ihrer Organisation gespeichert sind. Laden Sie in diesem Prozess das Metadatendokument hoch, das von der Identitätsanbieter-Software Ihrer Organisation wie im vorherigen Abschnitt erwähnt erstellt wurde. Details hierzu finden Sie unter Erstellen Sie einen SAML-Identitätsanbieter in IAM.

Konfigurieren Sie die Berechtigungen AWS für Ihre Verbundbenutzer

Im nächsten Schritt erstellen Sie eine IAM-Rolle, die eine Vertrauensbeziehung zwischen Ihrer IAM und der IdP Ihrer Organisation einrichtet. Diese Rolle muss Ihren Identitätsanbieter als Auftraggeber

(vertrauenswürdige Entität) für die Zwecke des Identitätsverbunds identifizieren. Die Rolle definiert auch, was Benutzer, die vom IdP Ihrer Organisation authentifiziert wurden, tun dürfen. AWS Sie können diese Rolle mit der IAM-Konsole erstellen. Beim Erstellen der Vertrauensrichtlinie, die angibt, wer die Rolle übernehmen kann, geben Sie den SAML-Anbieter an, den Sie zuvor in IAM erstellt haben. Sie geben auch ein oder mehrere SAML-Attribute an, denen ein Benutzer entsprechen muss, um die Rolle übernehmen zu dürfen. Sie können beispielsweise festlegen, dass sich nur Benutzer, deren SAML-`eduPersonOrgDN`-Wert `ExampleOrg` lautet, anmelden können. Der Rollenassistent fügt automatisch eine Bedingung hinzu, um das Attribut `saml:aud` zu testen und sicherzustellen, dass die Rolle nur zur Anmeldung bei der AWS Management Console verwendet wird. Die Vertrauensrichtlinie für die Rolle kann wie folgt aussehen:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {"StringEquals": {
      "saml:edupersonorgdn": "ExampleOrg",
      "saml:aud": "https://signin.aws.amazon.com/saml"
    }}
  }]
}
```

> **Note**
>
> SAML-IDPs, die in einer Rollen-Vertrauensrichtlinie verwendet werden, müssen sich in demselben Konto befinden, in dem sich die Rolle befindet.

Sie können unter `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` regionale Endpunkte für das `saml:aud`-Attribut einfügen. Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte).

Für die Berechtigungsrichtlinie in der Rolle geben Sie Berechtigungen wie für beliebige andere Rollen, Benutzer oder Gruppen auch an. Wenn Benutzer in Ihrer Organisation beispielsweise Amazon EC2-Instances verwalten dürfen, erlauben Sie in der Berechtigungsrichtlinie ausdrücklich

Amazon EC2-Aktionen. Weisen Sie dafür eine verwaltete Richtlinie zu, beispielsweise die verwaltete Richtlinie Amazon EC2 Full Access.

Weitere Informationen zum Erstellen von Rollen für SAML-Identitätsanbieter finden Sie unter Eine Rolle für den SAML 2.0-Verbund erstellen (Konsole).

Konfiguration beenden und SAML-Zusicherungen erstellen

Informieren Sie Ihren SAML-IdP, AWS Ihren Dienstanbieter, indem Sie die `saml-metadata.xml` Datei installieren, die Sie unter `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` oder finden. `https://signin.aws.amazon.com/static/saml-metadata.xml` Eine Liste möglicher Werte für *`region-code`* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte).

Wie Sie diese Datei installieren, ist abhängig von Ihrem Identitätsanbieter. Bei einigen Anbietern können Sie die URL eingeben, woraufhin der Identitätsanbieter die Datei für Sie abruft und installiert. Bei anderen Anbietern müssen Sie die Datei über eine URL herunterladen und dann als lokale Datei bereitstellen. Eine Anleitung finden Sie in der Dokumentation Ihres Identitätsanbieters. Unter Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS finden Sie Links zur Onlinedokumentation für viele unterstützte SAML-Anbieter.

Konfigurieren Sie auch die Informationen, die der Identitätsanbieter als SAML-Attribute an AWS als Teil der Authentifizierungsantwort übergeben soll. Die meisten dieser Informationen werden in AWS Form von Zustandskontextschlüsseln angezeigt, die Sie in Ihren Richtlinien auswerten können. Diese Bedingungsschlüssel stellen sicher, dass nur autorisierte Benutzer in den richtigen Kontexten Berechtigungen für den Zugriff auf Ihre AWS -Ressourcen erhalten. Sie können Zeitfenster angeben, die die Zeiten, in denen die Konsole verwendet werden darf, einschränken. Sie können auch die maximale Zeit (bis zu 12 Stunden) angeben, die Benutzer auf die Konsole zugreifen können, bevor sie ihre Anmeldeinformationen aktualisieren müssen. Details hierzu finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren.

Temporäre IAM Sicherheitsanmeldeinformationen

Sie können die AWS Security Token Service (AWS STS) verwenden, um vertrauenswürdige Benutzer zu erstellen und ihnen temporäre Sicherheitsanmeldeinformationen zur Verfügung zu stellen, mit denen der Zugriff auf Ihre AWS Ressourcen gesteuert werden kann. Temporäre Sicherheitsanmeldeinformationen funktionieren fast genauso wie die langfristigen Zugriffsschlüssel-Anmeldeinformationen, mit folgenden Unterschieden:

- Temporäre Sicherheitsanmeldeinformationen sind über einen kurzen Zeitraum gültig, wie der Name schon sagt. Sie können mit einer Gültigkeitsdauer von wenigen Minuten bis mehrere Stunden konfiguriert werden. Wenn die Anmeldeinformationen abgelaufen sind, werden sie nicht AWS mehr erkannt und es ist auch kein Zugriff mehr über API-Anfragen möglich, die mit ihnen gestellt wurden.
- Temporäre Sicherheitsanmeldeinformationen werden nicht mit dem Benutzer gespeichert, sondern auf Anforderung des Benutzers dynamisch generiert und bereitgestellt. Wenn (oder sogar bevor) die temporären Anmeldeinformationen ablaufen, kann der Benutzer neue Anmeldeinformationen anfordern, solange der anfordernde Benutzer weiterhin dazu berechtigt ist.

Daher haben temporäre Anmeldeinformationen die folgenden Vorteile gegenüber langfristigen Anmeldeinformationen:

- Sie müssen keine langfristigen AWS Sicherheitsnachweise verteilen oder in eine Anwendung einbetten.
- Sie können Benutzern Zugriff auf Ihre AWS Ressourcen gewähren, ohne eine AWS Identität für sie definieren zu müssen. Temporäre Anmeldeinformationen sind die Grundlage für Rollen und den Identitätsverbund.
- Die temporären Anmeldeinformationen haben eine begrenzte Nutzungsdauer. Somit müssen Sie sie aktualisieren oder explizit widerrufen, wenn Sie sie nicht mehr benötigen. Nachdem die temporären Anmeldeinformationen abgelaufen sind, können sie nicht erneut verwendet werden. Sie können die Gültigkeit der Anmeldeinformationen bis zu einem bestimmten Höchstwert festlegen.

AWS STS und AWS Regionen

Temporäre Sicherheitsanmeldeinformationen werden von AWS STS generiert. Standardmäßig AWS STS ist dies ein globaler Dienst mit einem einzigen Endpunkt bei`https://sts.amazonaws.com`. Sie können sich jedoch auch dafür entscheiden, AWS STS API-Aufrufe an Endpunkte in jeder anderen unterstützten Region zu tätigen. Dies kann die Latenz (Serververzögerung) verringern, indem die Anforderungen an Server in einer Region in Ihrer Nähe gesendet werden. Ihre Anmeldeinformationen sind unabhängig von der Region, in der sie generiert werden, weltweit gültig. Weitere Informationen finden Sie unter Verwaltung AWS STS in einem AWS-Region.

Gängige Szenarien für temporäre Anmeldeinformationen

Temporäre Anmeldeinformationen sind in Szenarien mit Identitätsverbund, Delegierung, kontoübergreifenden Zugriff und IAM-Rollen nützlich.

Identitätsverbund

Sie können Ihre Benutzeridentitäten in einem externen System außerhalb von verwalten AWS und Benutzern, die sich von diesen Systemen aus anmelden, Zugriff gewähren, um AWS Aufgaben auszuführen und auf Ihre AWS Ressourcen zuzugreifen. IAM unterstützt zwei Arten von Identitätsverbund. In beiden Fällen werden die Identitäten außerhalb von gespeichert. AWS Der Unterschied liegt darin, wo Ihr externes System angesiedelt ist &endash in Ihrem Rechenzentrum oder bei einem Anbieter im Internet. Weitere Informationen zu externen Identitätsanbietern erhalten Sie unter Identitätsanbieter und Verbund.

- SAML-Verbund — Sie können Benutzer im Netzwerk Ihrer Organisation authentifizieren und diesen Benutzern dann Zugriff darauf gewähren, AWS ohne neue AWS Identitäten für sie zu erstellen und von ihnen zu verlangen, dass sie sich mit anderen Anmeldeinformationen anmelden müssen. Dies wird als Single-Sign-On-Ansatz für temporären Zugriff bezeichnet. AWS STS unterstützt offene Standards wie Security Assertion Markup Language (SAML) 2.0, mit der Sie Microsoft AD FS verwenden können, um Ihr Microsoft Active Directory optimal zu nutzen. Sie können mit SAML 2.0 auch Ihre eigene Lösung zum Verbinden von Benutzeridentitäten verwalten. Weitere Informationen finden Sie unter SAML 2.0-Verbund.
  - Benutzerdefinierter Federation Broker — Sie können das Authentifizierungssystem Ihrer Organisation verwenden, um Zugriff auf Ressourcen zu gewähren. AWS Ein Beispielszenario finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.
  - Verbund mit SAML 2.0 – Verwenden Sie das Authentifizierungssystem Ihrer Organisation und SAML, um Zugriff auf AWS -Ressourcen zu gewähren. Weitere Informationen sowie ein Beispielszenario finden Sie unter SAML 2.0-Verbund.
- OpenID Connect (OIDC) -Verbund — Sie können es Benutzern ermöglichen, sich mit einem bekannten externen Identitätsanbieter wie Login with Amazon, Facebook, Google oder einem anderen OIDC 2.0-kompatiblen Anbieter für Ihre Mobil- oder Webanwendung anzumelden. Sie müssen keinen benutzerdefinierten Anmeldecode erstellen oder Ihre eigenen Benutzeridentitäten verwalten. Die Verwendung des OIDC-Verbunds hilft Ihnen dabei, Ihre AWS-Konto Sicherheit zu gewährleisten, da Sie keine langfristigen Sicherheitsanmeldedaten wie IAM-Benutzerzugriffsschlüssel mit Ihrer Anwendung verteilen müssen. Weitere Informationen finden Sie unter OIDC-Föderation.

AWS STS Der OIDC-Verbund unterstützt die Login with Amazon, Facebook, Google und jedem OpenID Connect (OIDC) -kompatiblen Identitätsanbieter.

> **Note**
>
> Für mobile Anwendungen empfehlen wir die Verwendung von Amazon Cognito. Sie können diesen Service mit AWS SDKs für die mobile Entwicklung verwenden, um eindeutige Identitäten für Benutzer zu erstellen und sie für den sicheren Zugriff auf Ihre Ressourcen zu authentifizieren. AWS Amazon Cognito unterstützt dieselben Identitätsanbieter wie AWS STS und unterstützt auch nicht authentifizierten (Gast-) Zugriff. Außerdem können Sie Benutzerdaten migrieren, wenn sich ein Benutzer anmeldet. Amazon Cognito stellt auch API-Operationen für die Synchronisierung von Benutzerdaten bereit, sodass diese erhalten bleiben, wenn die Benutzer zwischen Geräten wechseln. Weitere Informationen finden Sie unter Authentifizierung mit Amplify in der Amplify-Dokumentation.

Rollen für den kontoübergreifenden Zugriff

Viele Organisationen verfügen über mehrere AWS-Konto. Mithilfe von Rollen und dem kontoübergreifenden Zugriff können Sie Benutzeridentitäten in einem Konto definieren und diese Identitäten für den Zugriff auf AWS -Ressourcen in anderen Konten Ihrer Organisation verwenden. Dies wird als Delegierung für den temporären Zugriff bezeichnet. Weitere Informationen zum Erstellen kontenübergreifender Rollen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer. Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

IAM-Rollen für Amazon EC2

Wenn Sie Anwendungen auf Amazon EC2-Instances ausführen und diese Anwendungen auf AWS -Ressourcen zugreifen müssen, können Sie temporäre Sicherheitsanmeldeinformationen für Ihre Instances bereitstellen, wenn Sie sie starten. Diese temporären Sicherheitsanmeldeinformationen sind für alle Anwendungen verfügbar, die in der Instance ausgeführt werden. Sie müssen daher keine langfristigen Anmeldeinformationen in der Instance speichern. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden.

Andere Dienste AWS

Für den Zugriff auf die meisten AWS Dienste können Sie temporäre Sicherheitsanmeldedaten verwenden. Eine Liste der Services, die temporäre Sicherheitsanmeldeinformationen zulassen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

Anfordern temporärer Sicherheitsanmeldeinformationen

Um temporäre Sicherheitsanmeldedaten anzufordern, können Sie AWS Security Token Service (AWS STS) -Operationen in der AWS API verwenden. Dazu gehören Operationen zur Erstellung und Bereitstellung temporärer Sicherheitsanmeldedaten für vertrauenswürdige Benutzer, mit denen der Zugriff auf Ihre AWS Ressourcen gesteuert werden kann. Weitere Informationen zu finden AWS STS Sie unterTemporäre IAM Sicherheitsanmeldeinformationen. Weitere Informationen zu den verschiedenen Methoden, die Sie verwenden können, um temporäre Anmeldeinformationen anzufordern, indem Sie eine Rolle übernehmen, finden Sie unter Verwenden von IAM-Rollen.

Zum Aufrufen der API-Operationen können Sie eines der AWS -SDKs verwenden. Die SDKs sind für eine Vielzahl von Programmiersprachen und Umgebungen verfügbar, einschließlich Java, .NET, Python, Ruby, Android und iOS. Mithilfe der SDKs lassen sich Ihre Anforderungen kryptografisch signieren, Anforderungen bei Bedarf wiederholen und Fehlermeldungen verarbeiten. Sie können auch die AWS STS Query API verwenden, die in der AWS Security Token Service API-Referenz beschrieben wird. Schließlich unterstützen zwei Befehlszeilentools die AWS STS Befehle: the AWS Command Line Interface, und the AWS Tools for Windows PowerShell.

Die AWS STS API-Operationen erstellen eine neue Sitzung mit temporären Sicherheitsanmeldeinformationen, die ein Zugriffsschlüsselpaar und ein Sitzungstoken enthalten. Das Zugriffschlüsselpaar besteht aus einer Zugriffsschlüssel-ID und einem geheimen Schlüssel. Benutzer (oder eine Anwendung, die vom Benutzer ausgeführt wird) können diese Anmeldeinformationen verwenden, um auf Ihre Ressourcen zuzugreifen. Sie können eine Rollensitzung erstellen und Sitzungsrichtlinien und Sitzungs-Tags mithilfe von AWS STS API-Vorgängen programmgesteuert übergeben. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der Identität des Benutzers oder der auf die Rolle basierenden Richtlinien und der Sitzungssrichtlinien. Weitere Informationen zu Sitzungsrichtlinien finden Sie unter Sitzungsrichtlinien. Weitere Hinweise zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

> **Note**
>
> Die Größe des Sitzungstokens, das AWS STS API-Operationen zurückgeben, ist nicht festgelegt. Es wird ausdrücklich empfohlen, dass Sie keine Annahmen über die maximale

Größe machen. Die typische Tokengröße ist kleiner als 4096 Bytes, diese kann jedoch variieren.

Verwendung AWS STS mit AWS Regionen

Sie können AWS STS API-Aufrufe entweder an einen globalen Endpunkt oder an einen der regionalen Endpunkte senden. Wenn Sie einen Endpunkt näher bei Ihnen wählen, können Sie die Latenz verringern und verbessern Sie die Leistung Ihrer API-Aufrufe verbessern. Sie können Ihre Aufrufe auch an einen alternativen regionalen Endpunkt richten, wenn Sie nicht mehr mit dem ursprünglichen Endpunkt kommunizieren können. Wenn Sie eines der verschiedenen AWS SDKs verwenden, verwenden Sie diese SDK-Methode, um eine Region anzugeben, bevor Sie den API-Aufruf tätigen. Wenn Sie HTTP-API-Anforderungen manuell konstruieren, müssen Sie die Anforderung selbst an den korrekten Endpunkt richten. Weitere Informationen finden Sie im AWS STS -Abschnitt von Regionen und Endpunkte und Verwaltung AWS STS in einem AWS-Region.

Im Folgenden sind die API-Operationen aufgeführt, mit denen Sie temporäre Anmeldeinformationen für die Verwendung in Ihrer AWS Umgebung und Ihren Anwendungen abrufen können.

AssumeRole - Kontenübergreifende Delegation und Föderation durch einen benutzerdefinierten Identity Broker

Der `AssumeRole` API-Vorgang ist nützlich, um bestehenden IAM-Benutzern den Zugriff auf AWS Ressourcen zu ermöglichen, auf die sie noch keinen Zugriff haben. Beispielsweise benötigt der Benutzer möglicherweise Zugriff auf Ressourcen in einem anderen AWS-Konto. Es ist auch nützlich, um vorübergehend privilegierten Zugang zu erhalten – zum Beispiel, um eine Multi-Faktor-Authentifizierung (MFA) zu ermöglichen. Sie müssen diese API mit den vorhandenen Benutzeranmeldeinformationen aufrufen. Um zu erfahren, wer diesen Vorgang aufrufen kann, siehe Vergleich der AWS STS API-Operationen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer und Konfigurieren eines MFA-geschützten API-Zugriffs.

Dieser Aufruf muss mit gültigen AWS Sicherheitsanmeldedaten erfolgen. Wenn Sie diesen Aufruf machen, geben Sie die folgenden Informationen weiter:

- Den Amazon-Ressourcennamen (ARN) der Rolle, die die App übernehmen soll.
- (Optional) Die Dauer, die die Gültigkeitsdauer der temporären Sicherheitsanmeldeinformationen angibt. Verwenden Sie den `DurationSeconds`-Parameter, um die Dauer der Rollensitzung

von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle anzugeben. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle. Wenn Sie diesen Parameter nicht übergeben, laufen die temporären Anmeldeinformationen in einer Stunde ab. Der `DurationSeconds`-Parameter von dieser API ist unabhängig vom `SessionDuration`-HTTP-Parameter, den Sie verwenden, um die Dauer einer Konsolensitzung anzugeben. Verwenden Sie den `SessionDuration`-HTTP-Parameter in der Anforderung eines Konsolen-Anmelde-Tokens, die an den Verbund-Endpunkt gestellt wird. Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

- Rollensitzungsname. Verwenden Sie diesen Zeichenfolgenwert, um die Sitzung zu identifizieren, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird. Aus Sicherheitsgründen können Administratoren dieses Feld in AWS CloudTrail -Protokollen anzeigen, um festzustellen, wer eine Aktion in AWS ausgeführt hat. Ihr Administrator erfordert möglicherweise, dass Sie Ihren IAM-Benutzernamen als Sitzungsnamen angeben, wenn Sie die Rolle übernehmen. Weitere Informationen finden Sie unter `sts:RoleSessionName`.
- (Optional) Die Quellentität. Sie können festlegen, dass Benutzer eine Quellidentität angeben müssen, wenn sie eine Rolle übernehmen. Nachdem die Quellidentität festgelegt wurde, kann der Wert nicht geändert werden. Es ist in der Anforderung für alle Aktionen vorhanden, die während der Rollensitzung ausgeführt werden. Der Quellidentitätswert bleibt über Verketten von Rollen-Sitzungen. Sie können Informationen zur Quellenidentität in AWS CloudTrail Protokollen verwenden, um festzustellen, wer mit einer Rolle Aktionen ausgeführt hat. Weitere Informationen zur Verwendung von identitätsbasierten Richtlinien finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.
- (Optional) Eingebundene oder verwaltete Sitzungsrichtlinien. Diese Richtlinien beschränken die Berechtigungen der identitätsbasierten Richtlinie der Rolle, die der Rollensitzung zugeordnet sind. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie der Rolle, die angenommen wird, zulässig sind. Weitere Informationen über Rollensitzungsberechtigungen finden Sie unter Sitzungsrichtlinien.
- (Optional) Sitzungs-Tags. Sie können eine Rolle übernehmen und dann die temporären Anmeldeinformationen verwenden, um eine Anforderung zu stellen. Wenn Sie dies tun, enthalten die Auftraggebertags der Sitzung die Tags der Rolle und die übergebenen Sitzungs-Tags. Wenn Sie diesen Aufruf mit temporären Anmeldeinformationen tätigen, übernimmt die neue Sitzung auch transitive Sitzungs-Tags von der aufrufenden Sitzung. Weitere Hinweise zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

- (Optional) MFA-Informationen. Wenn Multi-Factor Authentication (MFA) verwendet werden kann, schließen Sie den Bezeichner für ein MFA-Gerät und den einmaligen Code ein, die von diesem Gerät bereitgestellt wurde.
- (Optional) Ein `ExternalId`-Wert, der verwendet werden kann, wenn der Zugriff auf Ihr Konto an einen Dritten delegiert wird. Durch diesen Wert wird sichergestellt, dass nur der angegebene Dritte auf die Rolle zugreifen kann. Weitere Informationen finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.

Im folgenden Beispiel wird eine Beispielanforderung und -antwort mit `AssumeRole` gezeigt. In dieser Beispielanforderung wird die `demo`-Rolle für die angegebene Dauer mit der enthaltenen Sitzungsrichtlinie, den Sitzungs-Tags und der externen ID und Quellidentität übernommen. Die resultierende Sitzung wird als `John-session` bezeichnet.

Example Beispielanforderung

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws::iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A
%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*%22%2C
%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

Der im vorherigen Beispiel gezeigte Richtlinienwert ist die URL-codierte Version der folgenden Richtlinie:

```
{"Version":"2012-10-17","Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

Der `AUTHPARAMS`-Parameter in diesem Beispiel ist ein Platzhalter für Ihre Signatur. Eine Signatur ist die Authentifizierungsinformation, die Sie AWS HTTP-API-Anfragen beifügen müssen. Wir empfehlen

die Verwendung der AWS -SDKs zum Erstellen von API-Anforderungen. Ein Vorteil dabei ist, dass die SDKs das Signieren von Anforderungen für Sie übernehmen. Wenn Sie API-Anfragen manuell erstellen und signieren müssen, finden Sie unter Signieren von AWS Anfragen mithilfe von Signature Version 4 in der Allgemeine Amazon Web Services-Referenz, wie Sie eine Anfrage signieren.

Zusätzlich zu den temporären Sicherheitsanmeldeinformationen, enthält die Antwort den Amazon-Ressourcennamen (ARN) für den Verbundbenutzer und die Ablaufzeit der Anmeldeinformationen.

Example Beispielantwort

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> Bei einer AWS Konvertierung werden die übergebenen Sitzungsrichtlinien und Sitzungs-Tags in ein gepacktes Binärformat komprimiert, für das ein separates Limit gilt. Ihre Anforderung

kann für diese Begrenzung fehlschlagen, selbst wenn Ihr Nur-Text-Inhalt die übrigen Anforderungen erfüllt. Das `PackedPolicySize`-Antwortelement gibt in Prozent an, wie nah die Richtlinien und Tags für Ihre Anforderung an der oberen Größengrenze liegen.

AssumeRoleWithWebIdentity - Verbund über einen webbasierten Identitätsanbieter

Die API-Operation `AssumeRoleWithWebIdentity` gibt einen Satz an temporären Sicherheitsanmeldeinformationen für Verbundbenutzer zurück, die über einen öffentlichen Identitätsanbieter authentifiziert werden. Beispiele für öffentliche Identitätsanbieter sind Login with Amazon, Facebook, Google oder ein beliebiger OpenID Connect-kompatibler Anbieter (OIDC). Dieser Vorgang ist nützlich, um mobile Anwendungen oder clientbasierte Webanwendungen zu erstellen, für die Zugriff auf erforderlich ist. AWS Wenn Sie diesen Vorgang verwenden, benötigen Ihre Benutzer keine eigenen Identitäten AWS oder IAM-Identitäten. Weitere Informationen finden Sie unter OIDC-Föderation.

Anstatt direkt anzurufen`AssumeRoleWithWebIdentity`, empfehlen wir, Amazon Cognito und den Amazon Cognito Credentials Provider mit den AWS SDKs für die mobile Entwicklung zu verwenden. Weitere Informationen finden Sie unter Authentifizierung mit Amplify in der Amplify-Dokumentation.

Wenn Sie Amazon Cognito nicht verwenden, rufen Sie die `AssumeRoleWithWebIdentity`-Aktion von AWS STS auf. Hierbei handelt es sich um einen nicht signierten Aufruf. Das bedeutet, dass die Anwendung keinen Zugriff auf AWS -Sicherheitsanmeldeinformationen haben muss, um den Aufruf zu senden. Wenn Sie diesen Aufruf machen, geben Sie die folgenden Informationen weiter:

- Den Amazon-Ressourcennamen (ARN) der Rolle, die die App übernehmen soll. Wenn Ihre App mehrere Möglichkeiten für das Anmelden der Benutzer unterstützt, müssen Sie mehrere Rollen definieren, eine pro Identitätsanbieter. Der Aufruf von `AssumeRoleWithWebIdentity` sollte die ARN der Rolle enthalten, die spezifisch für den Anbieter ist, über den der Benutzer sich angemeldet hat.
- Das Token, das die App vom Identitätsanbieter erhält, nachdem sie den Benutzer authentifiziert hat.
- Sie können Ihren IdP so konfigurieren, dass Attribute als Sitzungs-Tags an Ihr Token übergeben werden.
- (Optional) Die Dauer, die die Gültigkeitsdauer der temporären Sicherheitsanmeldeinformationen angibt. Verwenden Sie den `DurationSeconds`-Parameter, um die Dauer der Rollensitzung von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle anzugeben.

Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle. Wenn Sie diesen Parameter nicht übergeben, laufen die temporären Anmeldeinformationen in einer Stunde ab. Der `DurationSeconds`-Parameter von dieser API ist unabhängig vom `SessionDuration`-HTTP-Parameter, den Sie verwenden, um die Dauer einer Konsolensitzung anzugeben. Verwenden Sie den `SessionDuration`-HTTP-Parameter in der Anforderung eines Konsolen-Anmelde-Tokens, die an den Verbund-Endpunkt gestellt wird. Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

- Rollensitzungsname. Verwenden Sie diesen Zeichenfolgenwert, um die Sitzung zu identifizieren, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird. Aus Sicherheitsgründen können Administratoren dieses Feld in AWS CloudTrail -Protokollen anzeigen, um festzustellen, wer eine Aktion in AWS ausgeführt hat. Ihr Administrator erfordert möglicherweise, dass Sie einen bestimmten Wert für den Sitzungsnamen angeben, wenn Sie die Rolle übernehmen. Weitere Informationen finden Sie unter `sts:RoleSessionName`.
- (Optional) Die Quellentität. Sie können von Verbundbenutzern verlangen, dass sie eine Quellidentität angeben, wenn sie eine Rolle übernehmen. Nachdem die Quellidentität festgelegt wurde, kann der Wert nicht geändert werden. Es ist in der Anforderung für alle Aktionen vorhanden, die während der Rollensitzung ausgeführt werden. Der Quellidentitätswert bleibt über Verketten von Rollen-Sitzungen. Sie können Quellidentitätsinformationen in AWS CloudTrail Protokollen verwenden, um festzustellen, wer mit einer Rolle Maßnahmen ergriffen hat. Weitere Informationen zur Verwendung von identitätsbasierten Richtlinien finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.
- (Optional) Eingebundene oder verwaltete Sitzungsrichtlinien. Diese Richtlinien beschränken die Berechtigungen der identitätsbasierten Richtlinie der Rolle, die der Rollensitzung zugeordnet sind. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie der Rolle, die angenommen wird, zulässig sind. Weitere Informationen über Rollensitzungsberechtigungen finden Sie unter Sitzungsrichtlinien.

> **Note**
>
> Ein Aufruf an `AssumeRoleWithWebIdentity` ist nicht signiert (verschlüsselt). Daher sollten Sie diese optionale Sitzungsrichtlinie nur einschließen, wenn die Anforderung über einen vertrauenswürdigen Vermittler übertragen wird. In diesem Fall könnte jemand die Richtlinie ändern, um die Beschränkungen zu entfernen.

Wenn Sie anrufen`AssumeRoleWithWebIdentity`, AWS wird die Echtheit des Tokens überprüft. Je nach Anbieter AWS können Sie beispielsweise den Anbieter anrufen und das Token angeben, das die App übergeben hat. Unter der Annahme, dass der Identitätsanbieter das Token validiert, werden Ihnen die folgenden Informationen AWS zurückgegeben:

- Einen Satz temporärer Sicherheitsanmeldeinformationen. Diese bestehen aus einer Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sitzungs-Token.
- Die Rollen-ID und den ARN der übernommenen Rolle.
- Ein `SubjectFromWebIdentityToken`-Wert mit der eindeutigen Benutzer-ID.

Wenn Sie über die temporären Sicherheitsanmeldedaten verfügen, können Sie sie für AWS API-Aufrufe verwenden. Dies ist derselbe Vorgang wie das Durchführen eines AWS API-Aufrufs mit langfristigen Sicherheitsanmeldedaten. Der Unterschied besteht darin, dass Sie das Sitzungstoken einschließen müssen, mit dem AWS überprüfen kann, dass die temporären Sicherheitsanmeldeinformationen gültig sind.

Ihre App sollte die Anmeldeinformationen zwischenspeichern. Wie bereits erwähnt laufen die Anmeldeinformationen standardmäßig nach einer Stunde ab. Wenn Sie den CredentialsProviderAmazonSTS-Vorgang im AWS SDK nicht verwenden, liegt es an Ihnen und Ihrer App, `AssumeRoleWithWebIdentity` erneut aufzurufen. Rufen Sie diese Operation auf, um einen neuen Satz an temporären Sicherheitsanmeldeinformationen zu erhalten, bevor die alten ablaufen.

AssumeRoleWithSAML — Verbund über einen Enterprise Identity Provider, der mit SAML 2.0 kompatibel ist

Die API-Operation `AssumeRoleWithSAML` gibt einen Satz an temporären Sicherheitsanmeldeinformationen für Verbundbenutzer zurück, die über das bestehende Identitätssystem Ihrer Organisation authentifiziert werden. Die Benutzer müssen auch SAML 2.0 (Security Assertion Markup Language) verwenden, um Authentifizierungs- und Autorisierungsinformationen an AWS zu übergeben. Diese API-Operation ist hilfreich für Organisationen, die ihre Identitätssysteme (z. B. Windows Active Directory oder OpenLDAP) in Software integriert haben, die SAML-Assertionen produzieren können. Eine solche Integration bietet Informationen über Benutzeridentität und Berechtigungen (wie Active Directory Federation Services oder Shibboleth). Weitere Informationen finden Sie unter SAML 2.0-Verbund.

> **Note**
>
> Ein Aufruf an `AssumeRoleWithSAML` ist nicht signiert (verschlüsselt). Daher sollten Sie diese optionale Sitzungsrichtlinie nur einschließen, wenn die Anforderung über einen vertrauenswürdigen Vermittler übertragen wird. In diesem Fall könnte jemand die Richtlinie ändern, um die Beschränkungen zu entfernen.

Hierbei handelt es sich um einen nicht signierten Aufruf, was bedeutet, dass die Anwendung keinen Zugriff auf AWS -Sicherheitsanmeldeinformationen benötigt, um den Aufruf zu tätigen. Wenn Sie diesen Aufruf machen, geben Sie die folgenden Informationen weiter:

- Den Amazon-Ressourcennamen (ARN) der Rolle, die die App übernehmen soll.
- Den ARN des SAML-Anbieters, der in IAM erstellt wurde und den Identitätsanbieter beschreibt.
- Die in Basis-64 kodierte SAML-Assertion, die vom SAML-Identitätsanbieter in seiner Authentifizierungsantwort auf die Anmeldeanforderung von Ihrer App bereitgestellt wurde.
- Sie können Ihren IdP so konfigurieren, dass Attribute an Ihre SAML-Assertion als Sitzungs-Tagsübergeben werden.
- (Optional) Die Dauer, die die Gültigkeitsdauer der temporären Sicherheitsanmeldeinformationen angibt. Verwenden Sie den `DurationSeconds`-Parameter, um die Dauer der Rollensitzung von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle anzugeben. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle. Wenn Sie diesen Parameter nicht übergeben, laufen die temporären Anmeldeinformationen in einer Stunde ab. Der `DurationSeconds`-Parameter von dieser API ist unabhängig vom `SessionDuration`-HTTP-Parameter, den Sie verwenden, um die Dauer einer Konsolensitzung anzugeben. Verwenden Sie den `SessionDuration`-HTTP-Parameter in der Anforderung eines Konsolen-Anmelde-Tokens, die an den Verbund-Endpunkt gestellt wird. Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.
- (Optional) Eingebundene oder verwaltete Sitzungsrichtlinien. Diese Richtlinien beschränken die Berechtigungen der identitätsbasierten Richtlinie der Rolle, die der Rollensitzung zugeordnet sind. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie der Rolle, die angenommen wird, zulässig sind. Weitere Informationen über Rollensitzungsberechtigungen finden Sie unter Sitzungsrichtlinien.

- Rollensitzungsname. Verwenden Sie diesen Zeichenfolgenwert, um die Sitzung zu identifizieren, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird. Aus Sicherheitsgründen können Administratoren dieses Feld in AWS CloudTrail -Protokollen anzeigen, um festzustellen, wer eine Aktion in AWS ausgeführt hat. Ihr Administrator erfordert möglicherweise, dass Sie einen bestimmten Wert für den Sitzungsnamen angeben, wenn Sie die Rolle übernehmen. Weitere Informationen finden Sie unter `sts:RoleSessionName`.
- (Optional) Die Quellentität. Sie können von Verbundbenutzern verlangen, dass sie eine Quellidentität angeben, wenn sie eine Rolle übernehmen. Nachdem die Quellidentität festgelegt wurde, kann der Wert nicht geändert werden. Es ist in der Anforderung für alle Aktionen vorhanden, die während der Rollensitzung ausgeführt werden. Der Quellidentitätswert bleibt über Verketten von Rollen-Sitzungen. Sie können Quellidentitätsinformationen in AWS CloudTrail Protokollen verwenden, um festzustellen, wer mit einer Rolle Aktionen ausgeführt hat. Weitere Informationen zur Verwendung von identitätsbasierten Richtlinien finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

Wenn Sie anrufen`AssumeRoleWithSAML`, AWS wird die Echtheit der SAML-Assertion überprüft. Unter der Annahme, dass der Identitätsanbieter die Assertion validiert, werden Ihnen die folgenden AWS Informationen zurückgegeben:

- Einen Satz temporärer Sicherheitsanmeldeinformationen. Diese bestehen aus einer Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sitzungs-Token.
- Die Rollen-ID und den ARN der übernommenen Rolle.
- Einen `Audience`-Wert, der den Wert des `Recipient`-Attributs des `SubjectConfirmationData`-Elements der SAML-Zusicherung enthält.
- Einen `Issuer`-Wert, der den Wert des `Issuer`-Elements der SAML-Zusicherung enthält.
- Ein `NameQualifier` Element, das einen Hashwert enthält, der aus dem `Issuer` Wert, der AWS-Konto ID und dem Anzeigenamen des SAML-Anbieters besteht. In Kombination mit dem `Subject`-Element können sie den Verbundbenutzer eindeutig identifizieren.
- Ein `Subject`-Element, das den Wert des `NameID`-Elements im `Subject`-Element der SAML-Zusicherung enthält.
- Ein `SubjectType`-Element, das das Format des `Subject`-Elements angibt. Der Wert kann `persistent`, `transient`oder die vollständige `Format`-URI aus den `Subject`- und `NameID`-Elementen sein, die in Ihrer SAML-Zusicherung verwendet wurden. Weitere Informationen zum `NameID`-Attribut des `Format`-Elements finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren.

Wenn Sie über die temporären Sicherheitsanmeldedaten verfügen, können Sie sie für AWS API-Aufrufe verwenden. Dies ist derselbe Vorgang wie das Durchführen eines AWS API-Aufrufs mit langfristigen Sicherheitsanmeldedaten. Der Unterschied besteht darin, dass Sie das Sitzungstoken einschließen müssen, mit dem AWS überprüfen kann, dass die temporären Sicherheitsanmeldeinformationen gültig sind.

Ihre App sollte die Anmeldeinformationen zwischenspeichern. Standardmäßig laufen die Anmeldeinformationen nach einer Stunde ab. Wenn Sie die CredentialsProviderAmazonSTS-Aktion im AWS SDK nicht verwenden, liegt es an Ihnen und Ihrer App, `AssumeRoleWithSAML` erneut aufzurufen. Rufen Sie diese Operation auf, um einen neuen Satz an temporären Sicherheitsanmeldeinformationen zu erhalten, bevor die alten ablaufen.

GetFederationToken - Verbund durch einen benutzerdefinierten Identity Broker

Die API-Operation `GetFederationToken` gibt einen Satz an temporären Sicherheitsanmeldeinformationen für Verbundbenutzer zurück. Diese API unterscheidet sich von `AssumeRole` dahingehend, dass der Standard-Ablaufzeitraum wesentlich länger ist (zwölf Stunden anstelle von einer Stunde). Außerdem können Sie den `DurationSeconds`- Parameter verwenden, um eine Dauer anzugeben, in der die temporären Sicherheitsanmeldeinformationen gültig bleiben. Die resultierenden Anmeldeinformationen sind für die angegebene Dauer gültig, zwischen 900 Sekunden (15 Minuten) und 129.600 Sekunden (36 Stunden). Der längere Ablaufzeitraum kann dazu beitragen, die Anzahl der Anrufe zu reduzieren AWS , da Sie nicht so oft neue Anmeldeinformationen abrufen müssen.

Wenn Sie diese Anforderung tätigen, verwenden Sie die Anmeldeinformationen eines bestimmten IAM-Benutzers. Die Berechtigungen für die temporären Sicherheitsanmeldeinformationen werden von den Sitzungsrichtlinien bestimmt, die Sie beim Aufruf von `GetFederationToken` übergeben. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der IAM-Benutzerrichtlinien und der Sitzungsrichtlinien, die Sie übergeben. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie des IAM-Benutzers, der den Verbund anfordert, zulässig sind. Weitere Informationen über Rollensitzungsberechtigungen finden Sie unter Sitzungsrichtlinien.

Wenn Sie die temporären Anmeldeinformationen verwenden, die von der `GetFederationToken`-Operation zurückgegeben werden, enthalten die Auftraggeber-Tags der Sitzung die Tags des Benutzers und die übergebenen Sitzungs-Tags. Weitere Hinweise zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

Der `GetFederationToken`-Aufruf gibt temporäre Sicherheitsanmeldeinformationen zurück, die aus dem Sitzungstoken, dem Zugriffsschlüssel, dem geheimen Schlüssel und dem Ablaufdatum bestehen. Sie können `GetFederationToken` verwenden, wenn Sie Berechtigungen in Ihrer Organisation verwalten möchten (z. B. mithilfe der Proxy-Anwendung für die Zuweisung von Berechtigungen).

Im folgenden Beispiel wird eine Beispielanforderung und -antwort mit `GetFederationToken` gezeigt. In dieser Beispielanforderung wird der aufrufende Benutzer für die angegebene Dauer mit dem ARN der Sitzungsrichtlinie und den Sitzungs-Tagsverbunden. Die resultierende Sitzung wird als `Jane-session` bezeichnet.

Example Beispielanforderung

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

Der Richtlinien-ARN im vorherigen Beispiel enthält den folgenden URL-kodierten ARN:

```
arn:aws:iam::123456789012:policy/Role1policy
```

Beachten Sie außerdem, dass der `&AUTHPARAMS`-Parameter im Beispiel als Platzhalter für die Authentifizierungsinformationen gedacht ist. Dies ist die Signatur, die Sie AWS HTTP-API-Anfragen beifügen müssen. Wir empfehlen die Verwendung der AWS -SDKs zum Erstellen von API-Anforderungen. Ein Vorteil dabei ist, dass die SDKs das Signieren von Anforderungen für Sie übernehmen. Wenn Sie API-Anfragen manuell erstellen und signieren müssen, erfahren Sie unter Signieren von AWS Anfragen mithilfe von Signature Version 4 in der Allgemeine Amazon Web Services-Referenz, wie Sie eine Anfrage signieren.

Zusätzlich zu den temporären Sicherheitsanmeldeinformationen, enthält die Antwort den Amazon-Ressourcennamen (ARN) für den Verbundbenutzer und die Ablaufzeit der Anmeldeinformationen.

Example Beispielantwort

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>4</PackedPolicySize>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> Bei einer AWS Konvertierung werden die übergebenen Sitzungsrichtlinien und Sitzungs-Tags in ein gepacktes Binärformat komprimiert, für das ein separates Limit gilt. Ihre Anforderung kann für diese Begrenzung fehlschlagen, selbst wenn Ihr Nur-Text-Inhalt die übrigen Anforderungen erfüllt. Das `PackedPolicySize`-Antwortelement gibt in Prozent an, wie nah die Richtlinien und Tags für Ihre Anforderung an der oberen Größengrenze liegen.

AWS empfiehlt, dass Sie Berechtigungen auf Ressourcenebene gewähren (wenn Sie beispielsweise eine ressourcenbasierte Richtlinie an einen Amazon S3 S3-Bucket anhängen). Sie können den Parameter weglassen. `Policy` Wenn Sie jedoch keine Richtlinie für den Verbundbenutzer

einschließen, werden über die temporären Sicherheitsanmeldeinformationen keine Berechtigungen erteilt. In diesem Fall müssen Sie Ressourcenrichtlinien verwenden, um dem Verbundbenutzer Zugriff auf Ihre AWS -Ressourcen zu gewähren.

Nehmen wir zum Beispiel an, Ihre AWS-Konto Nummer ist 111122223333 und Sie haben einen Amazon S3 S3-Bucket, auf den Sie Susan Zugriff gewähren möchten. Susans temporäre Sicherheitsanmeldeinformationen enthalten keine Richtlinie für den Bucket. In diesem Fall müssen Sie sicherstellen, dass der Bucket eine Richtlinie mit einem ARN hat, der mit Susans ARN übereinstimmt, z. B. `arn:aws:sts::111122223333:federated-user/Susan`.

GetSessionToken - vorläufige Anmeldedaten für Benutzer in nicht vertrauenswürdigen Umgebungen

Die API-Operation `GetSessionToken` gibt einen Satz an temporären Sicherheitsanmeldeinformationen an einen vorhandenen IAM-Benutzer zurück. Dies ist nützlich, um mehr Sicherheit zu bieten, z. B. AWS Anfragen nur zuzulassen, wenn MFA für den IAM-Benutzer aktiviert ist. Da die Anmeldedaten temporär sind, bieten sie erweiterte Sicherheit, wenn ein IAM-Benutzer über eine weniger sichere Umgebung auf Ihre Ressourcen zugreift. Beispiele für weniger sicheren Umgebungen sind ein mobiles Gerät oder ein Webbrowser. Weitere Informationen finden Sie unter Anfordern temporärer Sicherheitsanmeldeinformationen oder GetSessionTokenin der AWS Security Token Service API-Referenz.

Standardmäßig sind temporäre Sicherheitsanmeldeinformationen für einen IAM-Benutzer für maximal zwölf Stunden gültig. Sie können jedoch mit dem `DurationSeconds`-Parameter die Dauer beliebig zwischen 15 Minuten und 36 Stunden festlegen. Aus Sicherheitsgründen Root-Benutzer des AWS-Kontos ist ein Token für eine auf eine Dauer von einer Stunde beschränkt.

`GetSessionToken` gibt temporäre Sicherheitsanmeldeinformationen bestehend aus einem Sitzungstoken, einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel zurück. Im folgenden Beispiel wird eine Beispielanforderung und -antwort mit `GetSessionToken` gezeigt. Die Antwort umfasst auch die Ablaufzeit der temporären Sicherheitsanmeldeinformationen.

Example Beispielanforderung

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

Der AUTHPARAMS-Parameter in diesem Beispiel ist ein Platzhalter für Ihre Signatur. Eine Signatur ist die Authentifizierungsinformation, die Sie AWS HTTP-API-Anfragen beifügen müssen. Wir empfehlen die Verwendung der AWS -SDKs zum Erstellen von API-Anforderungen. Ein Vorteil dabei ist, dass die SDKs das Signieren von Anforderungen für Sie übernehmen. Wenn Sie API-Anfragen manuell erstellen und signieren müssen, erfahren Sie unter Signieren von AWS Anfragen mithilfe von Signaturen Version 4 in der Allgemeine Amazon Web Services-Referenz, wie Sie eine Anfrage signieren.

Example Beispielantwort

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

Optional kann die `GetSessionToken` Anfrage `TokenCode` Werte für die Überprüfung der AWS Multi-Faktor-Authentifizierung (MFA) enthalten`SerialNumber`. Wenn die angegebenen Werte gültig sind, werden temporäre Sicherheitsanmeldeinformationen AWS STS bereitgestellt, die den Status der MFA-Authentifizierung enthalten. Die temporären Sicherheitsanmeldedaten können dann für den Zugriff auf die MFA-geschützten API-Operationen oder AWS Websites verwendet werden, solange die MFA-Authentifizierung gültig ist.

Das folgende Beispiel zeigt eine `GetSessionToken`-Anforderung, die einen MFA-Verifizierungscode und die Seriennummer des Geräts enthält.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> Der Anruf an AWS STS kann an den globalen Endpunkt oder an einen der regionalen Endpunkte erfolgen, den Sie aktivieren. AWS-Konto Weitere Informationen finden Sie im AWS STS -Abschnitt von Regionen und Endpunkte.
> Der AUTHPARAMS-Parameter in diesem Beispiel ist ein Platzhalter für Ihre Signatur. Eine Signatur ist die Authentifizierungsinformation, die Sie AWS HTTP-API-Anfragen beifügen müssen. Wir empfehlen die Verwendung der AWS -SDKs zum Erstellen von API-Anforderungen. Ein Vorteil dabei ist, dass die SDKs das Signieren von Anforderungen für Sie übernehmen. Wenn Sie API-Anfragen manuell erstellen und signieren müssen, finden Sie unter Signieren von AWS Anfragen mithilfe von Signature Version 4 in der Allgemeine Amazon Web Services-Referenz, wie Sie eine Anfrage signieren.

Vergleich der AWS STS API-Operationen

In der folgenden Tabelle werden die Funktionen der API-Operationen verglichen AWS STS , bei denen temporäre Sicherheitsanmeldedaten zurückgegeben werden. Weitere Informationen zu den verschiedenen Methoden, die Sie verwenden können, um temporäre Anmeldeinformationen anzufordern, indem Sie eine Rolle übernehmen, finden Sie unter Verwenden von IAM-Rollen. Weitere Informationen zu den verschiedenen AWS STS API-Vorgängen, mit denen Sie Sitzungs-Tags übergeben können, finden Sie unterSitzungs-Tags übergeben AWS STS.

Vergleichen der API-Optionen

AWS STS API	Wer kann aufrufen?	Lebensdauer der Anmeldeinformationen (min | max | Standard)	MFA-Unterstützung[1]	Sitzungsrichtlinienunterstützung[2]	Einschränkungen bei den sich ergebenden temporären Anmeldeinformationen
AssumeRole	IAM-Benutzer oder IAM-Rolle mit vorhandenen temporären Sicherheitsanmeldeinformationen	15 Min. | Maximale Sitzungsdauer[3] | 1 Std.	Ja	Ja	Kein Aufruf von `GetFederationToken` oder `GetSessionToken` .
AssumeRoleWithSAML	Jeder user;-Anrufer muss eine SAML-Authentifizierungsantwort übergeben, die die Authentifizierung von einem bekannten Identitätsanbieter angibt	15 Min. | Maximale Sitzungsdauer[3] | 1 Std.	Nein	Ja	Kein Aufruf von `GetFederationToken` oder `GetSessionToken` .
AssumeRoleWithWebIdentity	Jeder Benutzer; Anrufer muss ein OIDC-konformes JWT-Token übergeben, das auf die Authentifizierung durch einen bekannten	15 Min. | Maximale Sitzungsdauer[3] | 1 Std.	Nein	Ja	Kein Aufruf von `GetFederationToken` oder `GetSessionToken` .

AWS STS API	Wer kann aufrufen?	Lebensdauer der Anmeldeinformationen (min | max | Standard)	MFA-Unterstützung[1]	Sitzungsrichtlinienunterstützung[2]	Einschränkungen bei den sich ergebenden temporären Anmeldeinformationen
	Identitätsanbieter hinweist				
GetFederationToken	IAM-Benutzer oder Root-Benutzer des AWS-Kontos	IAM-Benutzer: 15 m | 36 h | 12 h Stammbenutzer: 15 Min. | 1 Stunde | 1 Stunde	Nein	Ja	IAM-Operationen können nicht mit der AWS CLI oder AWS API aufgerufen werden. Diese Beschränkung gilt nicht für Konsolensitzungen. Es können keine AWS STS Operationen außer `GetCallerIdentity` .,4 aufgerufen werden SSO an Konsole ist erlaubt.[5]
GetSessionToken	IAM-Benutzer oder Root-Benutzer des AWS-Kontos	IAM-Benutzer: 15 m | 36 h | 12 h Stammbenutzer: 15 Min. | 1 Stunde | 1 Stunde	Ja	Nein	IAM-API-Operationen können nur aufgerufen werden, wenn MFA-Informationen in der Anforderung enthalten sind. AWS STS API-Operationen außer `AssumeRole` oder `GetCallerIdentity` können nicht aufgerufen werden. SSO an Konsole ist nicht erlaubt.[6]

[1] MFA-Unterstützung. Sie können Informationen über ein Gerät mit Multi-Faktor-Authentifizierung (MFA) angeben, wenn Sie die AssumeRole und GetSessionToken API-Operationen aufrufen. Auf diese Weise wird sichergestellt, dass die temporären Sicherheitsanmeldeinformationen, die sich aus dem API-Aufruf ergeben, nur von Benutzern verwendet werden, die mit einem MFA-Gerät authentifiziert werden. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs.

[2] Sitzungsrichtlinienunterstützung. Sitzungsrichtlinien sind Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer programmgesteuert erstellen. Diese Richtlinie beschränkt die Berechtigungen der identitätsbasierten Richtlinie der Rolle oder des Benutzers, die bzw. der der Sitzung zugeordnet ist. Die resultierenden Sitzungsberechtigungen bilden die Schnittmenge der auf der Identität der Entity basierenden Richtlinien und der Sitzungsrichtlinien. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie der Rolle, die angenommen wird, zulässig sind. Weitere Informationen über Rollensitzungsberechtigungen finden Sie unter Sitzungsrichtlinien.

[3] Einstellung der maximalen Sitzungsdauer. Verwenden Sie den `DurationSeconds`-Parameter, um die Dauer Ihrer Rollensitzung von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle anzugeben. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.

x4. GetCallerIdentity Für diese Operation sind keine Berechtigungen erforderlich. Wenn ein Administrator Ihrem IAM-Benutzer oder Ihrer IAM-Rolle eine Richtlinie hinzufügt, die den Zugriff auf die `sts:GetCallerIdentity`-Aktion explizit verweigert, können Sie diese Operation trotzdem ausführen. Berechtigungen sind nicht erforderlich, da dieselben Informationen zurückgegeben werden, wenn einem IAM-Benutzer oder einer -Rolle der Zugriff verweigert wird. Eine Beispielantwort finden Sie unter Ich bin nicht berechtigt, Folgendes auszuführen: iam: MFADevice DeleteVirtual.

[5] Single Sign-On (SSO) an der Konsole. Um SSO zu unterstützen, AWS können Sie einen Verbundendpunkt (`https://signin.aws.amazon.com/federation`) aufrufen und temporäre Sicherheitsanmeldedaten übergeben. Der Endpunkt gibt ein Token zurück, das Sie verwenden können, um eine URL zu erstellen, die einen Benutzer direkt an der Konsole anmeldet, ohne dass ein Passwort erforderlich ist. Weitere Informationen finden Sie unter Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console und So aktivieren Sie den kontoübergreifenden Zugriff auf die AWS Managementkonsole im AWS Sicherheitsblog.

Nachdem Sie Ihre temporären Anmeldeinformationen abgerufen haben, können Sie nicht mehr auf die zugreifen, AWS Management Console indem Sie die Anmeldeinformationen an den Single Sign-

On-Endpunkt des Verbunds weitergeben. Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

Verwenden temporärer Anmeldeinformationen mit AWS -Ressourcen

Sie können temporäre Sicherheitsanmeldedaten verwenden, um programmatische Anfragen nach AWS Ressourcen mithilfe der AWS API AWS CLI oder (mithilfe der SDKs) zu stellen.AWS Die temporären Anmeldeinformationen bieten die gleichen Berechtigungen wie langfristige Sicherheitsanmeldeinformationen, z. B. IAM-Benutzer-Anmeldeinformationen. Es gibt jedoch einige Unterschiede:

- Wenn Sie einen Anruf mit temporären Sicherheitsanmeldeinformationen tätigen, muss der Aufruf ein Sitzungstoken enthalten, das zusammen mit diesen temporären Anmeldeinformationen zurückgegeben wird. AWS verwendet das Sitzungstoken, um die temporären Sicherheitsanmeldedaten zu validieren.
- Die temporären Anmeldeinformationen laufen nach einem bestimmten Intervall ab. Nach Ablauf der temporären Anmeldeinformationen schlagen alle Aufrufe, die Sie mit diesen Anmeldeinformationen tätigen, fehl, so dass Sie einen neuen Satz temporärer Anmeldeinformationen generieren müssen. Temporäre Anmeldeinformationen können nicht über das ursprünglich angegebene Intervall hinaus erweitert oder aktualisiert werden.
- Wenn Sie zur Erstellung einer Anforderung temporäre Anmeldeinformationen verwenden, enthält Ihr Auftraggeber möglicherweise eine Reihe von Tags. Diese Tags stammen von Sitzungs-Tags und Tags, die der von Ihnen angenommenen Rolle angefügt sind. Weitere Hinweise zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

Wenn Sie die AWS SDKs, die AWS Command Line Interface(AWS CLI) oder die Tools für Windows verwenden, unterscheidet sich die Art und Weise PowerShell, wie temporäre Sicherheitsanmeldeinformationen abgerufen und verwendet werden, je nach Kontext. Wenn Sie Code- oder Tools for PowerShell Windows-Befehle innerhalb einer EC2-Instance ausführen, können Sie Rollen für Amazon EC2 nutzen. AWS CLI Andernfalls können Sie eine AWS STS -API aufrufen, um die temporären Anmeldeinformationen abzurufen, und sie dann explizit verwenden, um Aufrufe an AWS -Services auszuführen.

> **Note**
>
> Sie können AWS Security Token Service (AWS STS) verwenden, um vertrauenswürdige Benutzer zu erstellen und ihnen temporäre Sicherheitsanmeldedaten zur Verfügung

zu stellen, mit denen der Zugriff auf Ihre AWS Ressourcen gesteuert werden kann. Weitere Informationen zu finden AWS STS Sie unterTemporäre IAM Sicherheitsanmeldeinformationen. AWS STS ist ein globaler Dienst mit einem Standardendpunkt beihttps://sts.amazonaws.com. Dieser Endpunkt befindet sich zwar in der Region USA Ost (Nord-Virginia), die Anmeldedaten, die Sie von diesem und anderen Endpunkten erhalten, sind jedoch weltweit gültig. Diese Anmeldeinformationen funktionieren mit Services und Ressourcen in jeder Region. Sie können sich auch dafür entscheiden, AWS STS API-Aufrufe an Endpunkte in einer der unterstützten Regionen zu tätigen. Dies kann die Latenz verringern, indem die Anforderungen an Server in einer Region in Ihrer Nähe gesendet werden. Ihre Anmeldeinformationen sind unabhängig von der Region, in der sie generiert werden, weltweit gültig. Weitere Informationen finden Sie unter Verwaltung AWS STS in einem AWS-Region.

Inhalt

Verwenden von temporären Anmeldeinformationen in Amazon EC2-Instances

Wenn Sie AWS CLI Befehle oder Code innerhalb einer EC2-Instance ausführen möchten, empfiehlt es sich, Rollen für Amazon EC2 zu verwenden, um Anmeldeinformationen abzurufen. Sie erstellen eine IAM-Rolle, die die Berechtigungen angibt, die Sie Anwendungen zuweisen möchten, die auf EC2-Instances ausgeführt werden. Wenn Sie die Instance starten, weisen Sie die Rolle der Instance zu.

Anwendungen und Tools for PowerShell Windows-Befehle AWS CLI, die auf der Instance ausgeführt werden, können dann automatische temporäre Sicherheitsanmeldedaten aus den Instance-Metadaten abrufen. Sie müssen die temporären Anmeldeinformationen nicht explizit abrufen. Die AWS SDKs und Tools für Windows rufen die Anmeldeinformationen PowerShell automatisch vom EC2 Instance Metadata Service (IMDS) ab und verwenden sie. AWS CLI Die temporären Anmeldeinformationen verfügen über die Berechtigungen, die Sie für die Rolle definieren, die mit der Instance verknüpft ist.

Weitere Informationen und Beispiele finden Sie in den folgenden Themen:

- Verwenden von IAM-Rollen, um Zugriff auf AWS Ressourcen in Amazon Elastic Compute Cloud zu gewähren — AWS SDK for Java
- Zugriff mithilfe einer IAM-Rolle gewähren — AWS SDK for .NET
- Erstellen einer IAM-Rolle - AWS SDK for Ruby

Verwenden von temporären Sicherheitsanmeldeinformationen mit den AWS -SDKs

Um temporäre Sicherheitsanmeldedaten im Code zu verwenden, rufen Sie programmgesteuert eine AWS STS API auf `AssumeRole` und extrahieren die resultierenden Anmeldeinformationen und das Sitzungstoken. Anschließend verwenden Sie diese Werte als Anmeldeinformationen für nachfolgende Aufrufe von. AWS Das folgende Beispiel zeigt Pseudocode für die Verwendung temporärer Sicherheitsanmeldedaten, wenn Sie ein AWS SDK verwenden:

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Ein Beispiel, das in Python geschrieben wurde (mit derAWS SDK for Python (Boto)), finden Sie unter Zu einer IAM-Rolle (AWS API) wechseln. In diesem Beispiel wird veranschaulicht, wie Sie mit `AssumeRole` temporäre Anmeldeinformationen abrufen und dann diese Anmeldeinformationen verwenden, um einen Aufruf von Amazon S3 durchzuführen.

Einzelheiten über den Aufruf von `AssumeRole`, `GetFederationToken` und anderen API-Vorgängen finden Sie in der AWS Security Token Service -API-Referenz. Informationen dazu, wie die temporären Sicherheitsanmeldeinformationen und das Sitzungstoken aus dem Ergebnis abgerufen werden, finden Sie in der Dokumentation für das SDK, mit dem Sie arbeiten. Sie finden die Dokumentation für alle AWS SDKs auf der AWS Hauptdokumentationsseite im Abschnitt SDKs und Toolkits.

Sie müssen sicherstellen, dass Sie neue Anmeldeinformationen erhalten, bevor die alten ablaufen. In einigen SDKs können Sie einen Anbieter verwenden, der die Aktualisierung der Anmeldeinformationen für Sie verwaltet. Sehen Sie in der Dokumentation für das von Ihnen verwendete SDK nach.

Verwenden von temporären Sicherheitsanmeldeinformationen mit der AWS CLI

Sie können temporäre Sicherheitsanmeldeinformationen mit der AWS CLI verwenden. Dies kann nützlich für das Testen von Richtlinien sein.

Unter Verwendung der AWS CLI können Sie eine AWS STS -API wie `AssumeRole` oder `GetFederationToken` aufrufen und dann die resultierende Ausgabe erfassen. Das folgende Beispiel zeigt einen Aufruf an `AssumeRole`, durch den die Ausgabe an eine Datei gesendet wird. Im Beispiel wird davon ausgegangen, dass es sich bei dem `profile` Parameter um ein Profil in der AWS CLI Konfigurationsdatei handelt. Es wird auch davon ausgegangen, dass Anmeldeinformationen für einen IAM-Benutzer referenziert werden, der über Berechtigungen verfügt, um die Rolle zu übernehmen.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Wenn der Befehl abgeschlossen ist, können Sie die Zugriffsschlüssel-ID, den geheimen Zugriffsschlüssel und den Sitzungs-Token unabhängig von deren Routingziel extrahieren. Sie können dies entweder manuell oder mithilfe eines Skripts tun. Anschließend können Sie diese Werte Umgebungsvariablen zuweisen.

Wenn Sie AWS CLI Befehle ausführen, AWS CLI sucht der in einer bestimmten Reihenfolge nach Anmeldeinformationen — zuerst in den Umgebungsvariablen und dann in der Konfigurationsdatei. Nachdem Sie die temporären Anmeldeinformationen in Umgebungsvariablen eingegeben haben, werden diese Anmeldeinformationen daher standardmäßig AWS CLI verwendet. (Wenn Sie im Befehl einen `profile` Parameter angeben, werden die Umgebungsvariablen AWS CLI übersprungen. Stattdessen wird in AWS CLI der Konfigurationsdatei gesucht, sodass Sie die Anmeldeinformationen in den Umgebungsvariablen bei Bedarf überschreiben können.)

Das folgende Beispiel zeigt, wie Sie die Umgebungsvariablen für temporäre Sicherheitsanmeldedaten festlegen und dann einen AWS CLI Befehl aufrufen können. Da der AWS CLI Befehl keinen `profile` Parameter enthält, AWS CLI sucht der zuerst in Umgebungsvariablen nach Anmeldeinformationen und verwendet daher die temporären Anmeldeinformationen.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
```

```
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Verwenden von temporären Sicherheitsanmeldeinformationen mit API-Operationen

Wenn Sie direkte HTTPS-API-Anfragen an stellen AWS, können Sie diese Anfragen mit den temporären Sicherheitsanmeldedaten signieren, die Sie von AWS Security Token Service (AWS STS) erhalten. Dazu verwenden Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel, die Sie von erhalten AWS STS. Verwenden Sie dazu die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel auf dieselbe Weise, wie Sie langfristige Anmeldeinformationen zum Signieren einer Anforderung verwenden würden. Sie fügen Ihrer API-Anfrage auch das Sitzungstoken hinzu, von dem Sie es erhalten AWS STS. Fügen Sie den Sitzungs-Token zu einem HTTP-Header oder zu einem Abfragezeichenfolgeparameter mit dem Namen `X-Amz-Security-Token` hinzu. Fügen Sie den Sitzungs-Token zum HTTP-Header oder zum Abfragezeichenfolgeparameter hinzu, jedoch nicht zu beiden. Weitere Informationen zum Signieren von HTTPS-API-Anfragen finden Sie unter Signieren von AWS API-Anfragen in der Allgemeine AWS-Referenz.

Weitere Informationen

Weitere Informationen zur Verwendung AWS STS mit anderen AWS Diensten finden Sie unter den folgenden Links:

- Amazon S3. Siehe Anforderungen mit temporären IAM-Benutzeranmeldeinformationen stellen oder Anforderungen mit temporären Benutzeranmeldeinformationen im Verbund stellen im Benutzerhandbuch für Amazon Simple Storage Service.
- Amazon SNS. Weitere Informationen finden Sie unter Verwenden identitätsbasierter Richtlinien mit Amazon SNS im Amazon Simple Notification Service Developer Guide.
- Amazon SQS. Weitere Informationen finden Sie unter Identitäts- und Zugriffsverwaltung in Amazon SQS im Amazon Simple Queue Service Developer Guide.
- Amazon SimpleDB. Siehe Using Temporary Security Credentials im Developer Guide für Amazon SimpleDB.

Steuern von Berechtigungen für temporäre Sicherheitsanmeldeinformationen

Sie können AWS Security Token Service (AWS STS) verwenden, um temporäre Sicherheitsanmeldeinformationen für vertrauenswürdige Benutzer zu erstellen und ihnen zur Verfügung zu stellen, mit denen der Zugriff auf Ihre AWS Ressourcen gesteuert werden kann. Weitere Informationen zu finden AWS STS Sie unterTemporäre IAM Sicherheitsanmeldeinformationen. Nachdem AWS STS temporäre Sicherheitsanmeldeinformationen ausgestellt hat, sind diese bis zum Ablaufzeitpunkt gültig und können nicht aufgehoben werden. Die den temporären Sicherheitsanmeldeinformationen zugewiesenen Berechtigungen werden jedoch bei jeder Anforderung, die über diese Anmeldeinformationen gestellt wird, erneut überprüft. daher können Sie die Gültigkeit von Anmeldeinformationen aufheben, indem Sie die Zugriffsrechte nach dem Ausstellen der Anmeldeinformationen widerrufen.

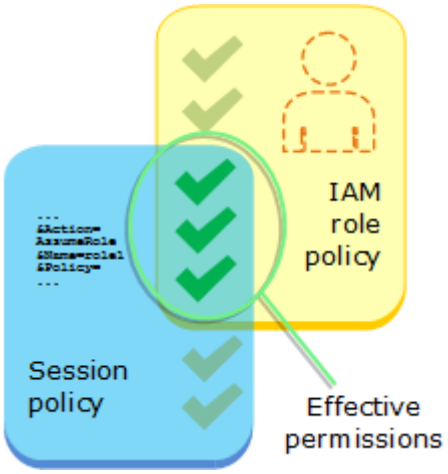
Bei den folgenden Themen wird davon ausgegangen, dass Sie über fundierte Kenntnisse im Bereich AWS Berechtigungen und Richtlinien verfügen. Weitere Informationen zu diesen Themen finden Sie unter Zugriffsmanagement für AWS Ressourcen.

Themen

Berechtigungen für AssumeRole, AssumeRoleWith SAML und AssumeRoleWithWebIdentity

Die Berechtigungsrichtlinie der übernommenen Rolle bestimmt die Berechtigungen für die temporären Sicherheitsanmeldeinformationen, die von `AssumeRole`, `AssumeRoleWithSAML` und `AssumeRoleWithWebIdentity` zurückgegeben werden. Sie definieren diese Berechtigungen beim Erstellen oder Aktualisieren der Rolle.

Optional können Sie eingebundene oder verwaltete Sitzungsrichtlinien als Parameter der API-Operationen `AssumeRole`, `AssumeRoleWithSAML` oder `AssumeRoleWithWebIdentity` übergeben. Sitzungsrichtlinien beschränken die Berechtigungen für die temporäre Anmeldedatensitzung der Rolle. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Sie können die temporären Anmeldeinformationen der Rolle in nachfolgenden AWS API-Aufrufen verwenden, um auf Ressourcen in dem Konto zuzugreifen, dem die Rolle gehört. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie der Rolle, die angenommen wird, zulässig sind. Weitere Informationen darüber, wie AWS die effektiven Berechtigungen einer Rolle bestimmt, finden Sie unter Auswertungslogik für Richtlinien.



Die Richtlinien, die an die Anmeldeinformationen angehängt sind, die den ursprünglichen Aufruf getätigt haben, `AssumeRole` werden AWS bei der Entscheidung über die Autorisierung „Zulassen“ oder „Verweigern“ nicht berücksichtigt. Der Benutzer verliert vorübergehend seine ursprünglichen Berechtigungen zugunsten der Berechtigungen, die von der angenommenen Rolle zugewiesen sind. Bei den Operationen `AssumeRoleWithSAML` und der `AssumeRoleWithWebIdentity` API gibt es keine zu bewertenden Richtlinien, da es sich bei dem API-Aufrufer nicht um eine AWS Identität handelt.

Beispiel: Zuweisen von Berechtigungen mit AssumeRole

Sie können die API-Operation `AssumeRole` mit verschiedenen Arten von Richtlinien verwenden. Nachfolgend sind einige Beispiele aufgeführt.

Rollenberechtigungsrichtlinie

In diesem Beispiel rufen Sie die API-Operation `AssumeRole` auf, ohne die Sitzungsrichtlinie im optionalen Parameter `Policy` anzugeben. Die den temporären Anmeldeinformationen zugewiesenen

Berechtigungen werden laut der Berechtigungsrichtlinie der übernommenen Rolle bestimmt. Das folgende Beispiel einer Berechtigungsrichtlinie gewährt der Rolle die Berechtigung, alle Objekte in einem S3-Bucket namens `productionapp` aufzulisten. Außerdem wird der Rolle gestattet, Objekte im Bucket zu platzieren, daraus abzurufen oder zu löschen.

Example Rollenberechtigungsrichtlinie

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Als Parameter übergebene Sitzungsrichtlinie

Stellen Sie sich vor, Sie möchten einem Benutzer erlauben, die gleiche Rolle wie im vorherigen Beispiel anzunehmen. Aber in diesem Fall soll die Rollensitzung nur über die Berechtigung zum Abrufen und Speichern von Objekten im S3-Bucket `productionapp` verfügen. Sie wollen nicht zulassen, dass sie Objekte löschen. Eine Möglichkeit, dies zu erreichen, ist das Erstellen einer neuen Rolle und das Angeben der gewünschten Berechtigungen in der Berechtigungsrichtlinie dieser Rolle. Eine weitere Möglichkeit besteht darin, die `AssumeRole`-API aufzurufen und eine Sitzungsrichtlinie in den optionalen `Policy`-Parameter als Teil der API-Operation einzuschließen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Sie können mit Sitzungsrichtlinien nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie der Rolle, die angenommen wird, zulässig sind. Weitere Informationen über Rollensitzungsberechtigungen finden Sie unter Sitzungsrichtlinien.

Nachdem Sie die temporären Anmeldedaten der neuen Sitzung abgerufen haben, können Sie sie an den Benutzer übergeben, der diese Berechtigungen erhalten soll.

Stellen Sie sich beispielsweise vor, dass Sie die folgende Richtlinie als Parameter des API-Aufrufs übergeben wird. Die Person, die die Sitzung verwendet, hat Berechtigungen, um nur diese Aktionen durchzuführen:

- Liste aller Objekte im `productionapp`-Bucket.
- Rufen und legen Sie Objekte im `productionapp`-Bucket ab.

In der folgenden Sitzungsrichtlinie wird die Berechtigung `s3:DeleteObject` herausgefiltert und der übernommenen Sitzung wird die Berechtigung `s3:DeleteObject` nicht gewährt. Die Richtlinie legt die maximalen Berechtigungen für die Rollensitzung so fest, dass sie alle vorhandenen Berechtigungsrichtlinien für die Rolle überschreibt.

Example Mit API-Aufruf **`AssumeRole`** übergebene Sitzungsrichtlinie

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Ressourcenbasierte Richtlinie

Einige AWS Ressourcen unterstützen ressourcenbasierte Richtlinien, und diese Richtlinien bieten einen weiteren Mechanismus zur Definition von Berechtigungen, die sich auf temporäre Sicherheitsanmeldedaten auswirken. Nur einige Ressourcen, wie Amazon S3-Buckets, Amazon

SNS-Themen und Amazon SQS-Warteschlangen, unterstützen ressourcenbasierte Richtlinien. Das folgende Beispiel ist eine Erweiterung der vorherigen Beispiele, in dem ein S3-Bucket mit dem Namen `productionapp` verwendet wird. Die folgende Richtlinie ist zum Bucket angefügt.

Wenn Sie die folgende ressourcenbasierte Richtlinie an den `productionapp`-Bucket anfügen, wird allen Benutzern die Berechtigung zum Löschen von Objekten aus dem Bucket verweigert. (Weitere Informationen finden Sie im `Principal`-Element in der Richtlinie.) Dies umfasst alle Benutzer der angenommenen Rolle, auch wenn die Rollenberechtigungsrichtlinie die `DeleteObject`-Berechtigung erteilt. Eine explizite `Deny`-Anweisung hat immer Vorrang vor einer `Allow`-Anweisung.

Example Beispiel einer Bucket-Richtlinie

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Weitere Informationen darüber, wie mehrere Richtlinientypen kombiniert und bewertet werden AWS, finden Sie unter. Auswertungslogik für Richtlinien

Überwachen und Steuern von Aktionen mit übernommenen Rollen

Eine IAM-Rolle ist ein Objekt in IAM, dem Berechtigungen zugewiesen sind. Wenn Sie diese Rolle mit einer IAM-Identität oder einer Identität von außerhalb übernehmen AWS, erhalten Sie eine Sitzung mit den Berechtigungen, die der Rolle zugewiesen sind.

Wenn Sie Aktionen in ausführen AWS, können die Informationen über Ihre Sitzung protokolliert werden, AWS CloudTrail sodass Ihr Kontoadministrator sie überwachen kann. Administratoren können Rollen so konfigurieren, dass Identitäten eine benutzerdefinierte Zeichenfolge übergeben müssen, die die Person oder Anwendung identifiziert, die Aktionen in AWS durchführt. Diese Identitätsinformation wird als Quellidentität in AWS CloudTrail gespeichert. Wenn der Administrator die Aktivitäten in überprüft CloudTrail, kann er anhand der Informationen zur Quellenidentität feststellen, wer oder was Aktionen in Sitzungen mit angenommenen Rollen ausgeführt hat.

Nachdem eine Quellidentität festgelegt wurde, ist sie in Anfragen für alle AWS Aktionen enthalten, die während der Rollensitzung ausgeführt wurden. Der festgelegte Wert bleibt bestehen, wenn

eine Rolle verwendet wird, um über die AWS CLI AWS OR-API eine andere Rolle anzunehmen, was als Rollenverkettung bezeichnet wird. Der festgelegte Wert kann während der Rollensitzung nicht geändert werden. Administratoren können detaillierte Berechtigungen auf der Grundlage des Vorhandenseins oder des Werts der Quellidentität konfigurieren, um die AWS Aktionen, die mit gemeinsam genutzten Rollen ausgeführt werden, weiter zu kontrollieren. Sie können entscheiden, ob das Quellidentitätsattribut verwendet werden kann, ob es erforderlich ist und welcher Wert verwendet werden kann.

Die Art und Weise, wie Sie die Quellidentität verwenden, unterscheidet sich von Rollensitzungsnamen und Sitzungs-Tags auf wichtige Weise. Der Quellidentitätswert kann nicht geändert werden, nachdem er festgelegt wurde, und er bleibt für alle zusätzlichen Aktionen bestehen, die mit der Rollensitzung ausgeführt werden. So können Sie Sitzungs-Tags und Rollensitzungsnamen verwenden:

- Sitzungs-Tags – Sie können Sitzungs-Tags übergeben, wenn Sie eine Rolle übernehmen oder einen Benutzer föderieren. Sitzungs-Tags sind vorhanden, wenn eine Rolle angenommen wird. Sie können Richtlinien definieren, die Tag-Bedingungsschlüssel verwenden, um Ihren Auftraggeber basierend auf ihren Tags Berechtigungen zu erteilen. Anschließend können Sie die Anfragen CloudTrail zur Übernahme von Rollen oder zur Zusammenführung von Benutzern anzeigen. Weitere Informationen zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.
- Rollensitzungsname – Sie können den `sts:RoleSessionName`-Bedingungsschlüssel in einer Rollenvertrauensrichtlinie verwenden, um zu verlangen, dass Ihre Benutzer einen bestimmten Sitzungsnamen angeben, wenn sie eine Rolle übernehmen. Rollensitzungsname kann verwendet werden, um Rollensitzungen zu unterscheiden, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird. Weitere Informationen zum Namen der Rollensitzung finden Sie unter sts: RoleSessionName.

Es wird empfohlen, die Quellidentität zu verwenden, wenn Sie die Identität steuern möchten, die eine Rolle übernimmt. Die Quellenidentität ist auch nützlich für das CloudTrail Miningprotokoll, um festzustellen, wer die Rolle zur Ausführung von Aktionen verwendet hat.

Themen

- Einrichten für die Verwendung der Quellidentität
- Wissenswertes über die Quellidentität
- Zum Festlegen der Quellidentität erforderliche Berechtigungen
- Angeben einer Quellidentität bei der Übernahme einer Rolle

- Verwenden Sie die Quellidentität mit AssumeRole
- Verwendung der Quellidentität mit AssumeRoleWith SAML
- Verwenden Sie die Quellidentität mit AssumeRoleWithWebIdentity
- Steuern des Zugriffs mithilfe von Informationen zur Quell
- Quellenidentität anzeigen in CloudTrail

Einrichten für die Verwendung der Quellidentität

Die Art und Weise, wie Sie die Quellidentität verwenden, hängt von der Methode ab, die verwendet wird, wenn Ihre Rollen angenommen werden. Beispielsweise können Ihre IAM-Benutzer Rollen direkt übernehmen, indem sie die `AssumeRole` verwenden. Wenn Sie über Unternehmensidentitäten, auch Personalidentitäten genannt, verfügen, greifen diese möglicherweise mithilfe von auf Ihre AWS Ressourcen zu. `AssumeRoleWithSAML` Wenn Endbenutzer auf Ihre mobilen oder Webanwendungen zugreifen, tun sie dies möglicherweise über `AssumeRoleWithWebIdentity`. Im Folgenden finden Sie einen Überblick über den Workflow auf hoher Ebene, mit dem Sie verstehen, wie Sie die Verwendung von Quellidentitätsinformationen in Ihrer vorhandenen Umgebung einrichten können.

1. Konfigurieren von Testbenutzern und -rollen – Konfigurieren Sie mithilfe einer Vorproduktionsumgebung Testbenutzer und -rollen und konfigurieren Sie ihre Richtlinien so, dass eine Quellidentität festgelegt werden kann.

   Wenn Sie einen Identitätsanbieter (IdP) für Ihre Verbundidentitäten verwenden, konfigurieren Sie Ihren IdP so, dass ein Benutzerattribut Ihrer Wahl für die Quellidentität in der Assertion oder Token übergeben wird.
2. Nehmen Sie die Rolle an. – Testen Sie das Annehmen von Rollen und Übergeben einer Quellidentität mit den Benutzern und Rollen, die Sie zum Testen eingerichtet haben.
3. Überprüfung CloudTrail — Überprüfen Sie die Quellidentitätsinformationen für Ihre Testrollen in Ihren CloudTrail Protokollen.
4. Trainieren von Benutzern – Stellen Sie nach dem Test in Ihrer Vorproduktionsumgebung sicher, dass Ihre Benutzer wissen, wie sie die Quellidentitätsinformationen eingeben, falls erforderlich. Legen Sie einen Stichtag fest, an dem Ihre Benutzer eine Quellidentität in Ihrer Produktionsumgebung angeben müssen.
5. Konfigurieren von Produktionsrichtlinien – Konfigurieren Sie Ihre Richtlinien für Ihre Produktionsumgebung und fügen Sie sie dann Ihren Produktionsbenutzern und -rollen hinzu.

6. Aktivität überwachen — Überwachen Sie die Aktivitäten Ihrer Produktionsrollen mithilfe von CloudTrail Protokollen.

Wissenswertes über die Quellidentität

Beachten Sie bei der Arbeit mit der Quellenidentität folgende Punkte.

- Bei der Verwendung von Sitzungs-Tags müssen die Rollenvertrauensrichtlinien für alle Rollen, die mit einem Identitätsanbieter (IdP) verbunden sind, über die `sts:SetSourceIdentity`-Berechtigung verfügen. Bei Rollen, die diese Berechtigung in der Vertrauensrichtlinie nicht haben, schlägt der `AssumeRole*`-Vorgang fehl. Wenn Sie die Vertrauensrichtlinie für Rollen nicht für jede Rolle aktualisieren möchten, können Sie eine separate IdP-Instance zum Übergeben von Sitzungs-Tags verwenden. Fügen Sie dann die `sts:SetSourceIdentity`-Berechtigung nur den Rollen hinzu, die mit dem separaten IdP verbunden sind.
- Wenn eine Identität eine Quellidentität festlegt, wird die `sts:SourceIdentity`-Schlüssel in der Anforderung. Für nachfolgende Aktionen, die während der Rollensitzung ausgeführt werden, ist der Schlüssel `aws:SourceIdentity` in der Anfrage vorhanden. AWS kontrolliert den Wert der Quellidentität weder in den Schlüsseln `sts:SourceIdentity` noch `aws:SourceIdentity`. Wenn Sie eine Quellidentität benötigen, müssen Sie ein Attribut auswählen, das Ihre Benutzer oder IdP bereitstellen sollen. Aus Sicherheitsgründen müssen Sie sicherstellen, dass Sie steuern können, wie diese Werte bereitgestellt werden.
- Der Wert der Quell-Identität muss zwischen 2 und 64 Zeichen lang sein und darf nur alphanumerische Zeichen, Unterstriche und die folgenden Zeichen enthalten:., + = @ -(Bindestrich). Sie können keinen Tag-Schlüssel oder -Wert erstellen, der mit dem Text **`aws:`** beginnt. Dieses Präfix ist für den AWS internen Gebrauch reserviert.
- Die Quellidentitätsinformationen werden nicht erfasst, CloudTrail wenn ein AWS Dienst oder eine dienstbezogene Rolle eine Aktion im Namen einer föderierten Identität oder einer Belegschaftsidentität ausführt.

> **Important**
>
> Sie können nicht zu einer Rolle in der wechseln, für AWS Management Console die eine Quellidentität festgelegt werden muss, wenn die Rolle übernommen wird. Um eine solche Rolle anzunehmen, können Sie die AWS CLI AWS OR-API verwenden, um den `AssumeRole` Vorgang aufzurufen und den Quellidentitätsparameter anzugeben.

Zum Festlegen der Quellidentität erforderliche Berechtigungen

Zusätzlich zu der Aktion, die der API-Operation entspricht, muss in Ihrer Richtlinie die folgende reine Berechtigungsaktion enthalten sein:

```
sts:SetSourceIdentity
```

- Um eine Quellidentität anzugeben, müssen Auftraggeber (IAM-Benutzer und IAM-Rollen) über Berechtigungen für `sts:SetSourceIdentity` haben. Als Administrator können Sie dies in der Rollenvertrauensrichtlinie und in der Berechtigungsrichtlinie des Auftraggebers konfigurieren.
- Wenn Sie eine Rolle mit einer anderen Rolle übernehmen, was als Rollenverkettung bezeichnet wird, sind Berechtigungen für `sts:SetSourceIdentity` sowohl in der Berechtigungsrichtlinie des Auftraggebers, der die Rolle übernimmt, als auch in der Rollenvertrauensrichtlinie der Zielrolle erforderlich. Andernfalls schlägt die Operation fehl.
- Bei der Verwendung von Sitzungs-Tags müssen die Rollenvertrauensrichtlinien für alle Rollen, die mit einem Identitätsanbieter (IdP) verbunden sind, über die `sts:SetSourceIdentity`-Berechtigung verfügen. Der `AssumeRole*`-Vorgang schlägt für jede Rolle fehl, die mit einem IdP verbunden ist, der Sitzungs-Tags ohne diese Berechtigung übergibt. Wenn Sie die Vertrauensrichtlinie für Rollen nicht für jede Rolle aktualisieren möchten, können Sie eine separate IdP Instance zum Übergeben der Quell-Identität verwenden und die `sts:SetSourceIdentity`-Berechtigung nur die Rollen, die mit dem separaten IdP verbunden sind.
- Um eine Quellidentität über Kontogrenzen hinweg festzulegen, müssen Sie die Berechtigung `sts:SetSourceIdentity` an zwei Stellen einfügen. Sie muss sich in der Berechtigungsrichtlinie des Auftraggebers im Ursprungskonto und in der Rollenvertrauensrichtlinie der Rolle im Zielkonto befinden. Dies kann z. B. erforderlich sein, wenn eine Rolle verwendet wird, um eine Rolle in einem anderen Konto mit Rollenverkettung zu übernehmen.

Stellen Sie sich als Kontoadministrator vor, Sie möchten den IAM-Benutzer `DevUser` in Ihrem Konto, um die `Developer_Role` auf demselben Konto zu übernehmen. Sie möchten diese Aktion jedoch nur zulassen, wenn der Benutzer die Quellidentität auf seinen IAM-Benutzernamen festgelegt hat. Sie können dazu diesem Benutzer die folgende IAM-Richtlinie zuweisen.

Example Beispiel für eine identitätsbasierte Richtlinie, die angehängt ist DevUser

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
}
```

Um die zulässigen Werte für die Quellidentität zu erzwingen, können Sie die folgende Rollenvertrauensrichtlinie konfigurieren. Die Richtlinie gibt dem IAM-Benutzer `DevUser` die Berechtigung, die Rolle zu übernehmen und eine Quellidentität festzulegen. Der `sts:SourceIdentity`-Bedingungsschlüssel definiert den zulässigen Wert der Quellenidentität.

Example Beispiel einer Rollenvertrauensrichtlinie für Quellidentität

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
```

```
        }
      }
    }
  ]
}
```

Unter Verwendung der Anmeldeinformationen für den IAM-Benutzer versucht der Benutzer anzunehmen`DevUser`, dass er die folgende Anfrage `DeveloperRole` verwendet. AWS CLI

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Bei der AWS Auswertung der Anfrage enthält der Anforderungskontext das `sts:SourceIdentity` of`DevUser`.

Angeben einer Quellidentität bei der Übernahme einer Rolle

Sie können eine Quellidentität angeben, wenn Sie eine der AWS STS `AssumeRole*` API-Operationen verwenden, um temporäre Sicherheitsanmeldeinformationen für eine Rolle abzurufen. Die API-Operation, die Sie verwenden, unterscheidet sich je nach Anwendungsfall. Wenn Sie beispielsweise IAM-Rollen verwenden, um IAM-Benutzern Zugriff auf AWS Ressourcen zu gewähren, auf die sie normalerweise keinen Zugriff haben, können Sie den `AssumeRole` Vorgang verwenden. Wenn Sie zur Verwaltung Ihrer Mitarbeiter den Identitätsverbund verwenden, können Sie den Vorgang `AssumeRoleWithSAML` verwenden. Wenn Sie den OIDC-Verbund verwenden, um Endbenutzern den Zugriff auf Ihre Mobil- oder Webanwendungen zu ermöglichen, können Sie den Vorgang verwenden. `AssumeRoleWithWebIdentity` In den folgenden Abschnitten wird die Verwendung von Quellidentitäten bei jedem Vorgang erläutert. Weitere Informationen über häufige Szenarien für temporäre Berechtigungsnachweise finden Sie unter Gängige Szenarien für temporäre Anmeldeinformationen.

Verwenden Sie die Quellidentität mit AssumeRole

Der `AssumeRole` Vorgang gibt einen Satz temporärer Anmeldeinformationen zurück, mit denen Sie auf AWS Ressourcen zugreifen können. Sie können IAM-Benutzer- oder Rollenanmeldeinformationen verwenden, um `AssumeRole` aufzurufen. Verwenden Sie die `--`

`source-identity` AWS CLI Option oder den `SourceIdentity` AWS API-Parameter, um bei der Übernahme einer Rolle die Quellenidentität zu übergeben. Das folgende Beispiel zeigt, wie die Quellidentität mit der AWS CLI angegeben werden kann.

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

Verwendung der Quellidentität mit AssumeRoleWith SAML

Die `AssumeRoleWithSAML`-Operation wird mit dem SAML-basierten Verbund authentifiziert. Dieser Vorgang gibt einen Satz temporärer Anmeldeinformationen zurück, mit denen Sie auf AWS Ressourcen zugreifen können. Weitere Informationen zur Verwendung eines SAML-basierten Verbunds für den AWS Management Console Zugriff finden Sie unter. Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console Einzelheiten zum AWS CLI AWS API-Zugriff finden Sie unter. SAML 2.0-Verbund Eine Anleitung zur Einrichtung eines SAML-Verbunds für Ihre Active Directory-Benutzer finden Sie unter AWS Verbundauthentifizierung mit Active Directory Federation Services (ADFS) im AWS Sicherheitsblog.

Als Administrator können Sie es Mitgliedern Ihres Unternehmensverzeichnisses ermöglichen, den Vorgang gemeinsam zu AWS nutzen. AWS STS `AssumeRoleWithSAML` Hierfür müssen Sie die folgenden Aufgaben ausführen:

1. Konfigurieren Sie einen SAML-Anbieter in Ihrer Organisation.
2. Erstellen eines SAML-Anbieters in IAM.
3. Konfigurieren Sie eine Rolle und ihre Berechtigungen AWS für Ihre Verbundbenutzer.
4. Abschließen der Konfiguration des SAML-Identitätsanbieters und Erstellen von Zusicherungen für die SAML-Authentifizierungsantwort.

Um ein SAML-Attribut für die Quellidentität festzulegen, schließen Sie das `Attribute`-Element mit dem `Name`-Attribut gesetzt auf `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Verwenden Sie das `AttributeValue`-Element, um den Wert der Quellidentität anzugeben. Angenommen, Sie möchten die folgenden Identitätsattribute als Sitzungs-Tags übergeben.

`SourceIdentity:DiegoRamirez`

Um diese Attribute zu übergeben, schließen Sie die folgenden Elemente in Ihre SAML-Zusicherung ein.

Example Beispielausschnitt einer SAML-Zusicherung

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Verwenden Sie die Quellidentität mit AssumeRoleWithWebIdentity

Der Principal, der den `AssumeRoleWithWebIdentity` Vorgang aufruft, wird mithilfe eines OpenID Connect (OIDC) -kompatiblen Verbunds authentifiziert. Diese Operation gibt einen Satz temporärer Anmeldeinformationen zurück, die Sie für den Zugriff auf AWS -Ressourcen verwenden können. Weitere Hinweise zur Verwendung des OIDC-Verbunds für den Zugriff finden Sie unter. AWS Management Console OIDC-Föderation

Um die Quellidentität von OpenID Connect (OIDC) zu übergeben, müssen Sie die Quellidentität in das JSON Web Token (JWT) einbeziehen. Fügen Sie Sitzungs-Tags in den `https://aws.amazon.com/ source_identity`-Namespace in dem Token ein, wenn Sie die `AssumeRoleWithWebIdentity`-Anforderung senden. Weitere Informationen zu OIDC-Token und Ansprüchen finden Sie unter Verwenden von Token mit Benutzerpools im Amazon Cognito Developer Guide.

Der folgende entschlüsselte JWT ist beispielsweise ein Token, das zum Aufrufen von `AssumeRoleWithWebIdentity` mit der Quellidentität `Admin` verwendet wird.

Example Beispiel für ein dekodiertes JSON-Web-Token

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Steuern des Zugriffs mithilfe von Informationen zur Quell

Wenn eine Quellidentität anfänglich festgelegt wird, ist der SourceIdentity Schlüssel sts: in der Anforderung vorhanden. Nachdem eine Quellidentität festgelegt wurde, ist der SourceIdentity Schlüssel aws: in allen nachfolgenden Anfragen vorhanden, die während der Rollensitzung gestellt wurden. Als Administrator können Sie Richtlinien schreiben, die eine bedingte Autorisierung zur Ausführung von AWS Aktionen gewähren, die auf der Existenz oder dem Wert des Quellidentitätsattributs basieren.

Stellen Sie sich vor, Sie möchten von Ihren Entwicklern verlangen, dass sie eine Quellidentität festlegen, damit sie eine wichtige Rolle übernehmen können, die über Schreibberechtigungen für eine produktionskritische AWS Ressource verfügt. Stellen Sie sich außerdem vor, Sie gewähren AWS Zugriff auf die Identitäten Ihrer Belegschaft mithilfe von`AssumeRoleWithSAML`. Sie möchten nur, dass ältere Entwickler Saanvi und Diego Zugriff auf die Rolle haben. Daher erstellen Sie die folgende Vertrauensrichtlinie für die Rolle.

Example Beispiel einer Rollenvetrauensichtlinie für Quellidentität (SAML)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

Die Vertrauensrichtlinie enthält eine Bedingung für `sts:SourceIdentity`, die eine Quellidentität erfordert, die auf Saanvi oder Diego eingestellt ist, um die kritische Rolle anzunehmen.

Wenn Sie einen OIDC-Anbieter für den Verbund verwenden und Benutzer mit diesem authentifiziert sind`AssumeRoleWithWebIdentity`, könnte Ihre Rollenvertrauensrichtlinie alternativ wie folgt aussehen.

Example Beispiel einer Rollenvetrauensichtlinie für Quellidentität (OIDC-Anbieter)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
```

```
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

Rollenverkettung und kontenübergreifende Anforderungen

Stellen Sie sich vor, Sie möchten Benutzern, die `CriticalRole` angenommen haben, erlauben, ein `CriticalRole_2` in einem anderen Konto anzunehmen. Die Anmeldeinformationen für die Rollensitzung, die für die Annahme von `CriticalRole` erlangt wurden, werden für die Rollenkette einer zweiten Rolle, `CriticalRole_2`, in einem anderen Konto verwendet. Die Rolle wird über eine Kontengrenze hinweg übernommen. Daher muss die Berechtigung `sts:SetSourceIdentity` sowohl in der Berechtigungsrichtlinie für `CriticalRole` als auch in der Rollenvertrauensrichtlinie für `CriticalRole_2` erteilt werden.

Example Beispiel für eine Berechtigungsrichtlinie für CriticalRole

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Um das Festlegen der Quellidentität über die Kontengrenze hinweg zu sichern, vertraut die folgende Rollenvertrauensrichtlinie nur dem Rollenauftraggeber für`CriticalRole`, um die Quellidentität festzulegen.

Example Beispiel für eine Vertrauensrichtlinie für eine Rolle auf CriticalRole _2

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

Der Benutzer führt den folgenden Aufruf mit den Anmeldeinformationen für die Rollensitzung durch, die er bei der Annahme abgerufen hat CriticalRole. Die Quellidentität wurde während der Annahme von festgelegt CriticalRole, sodass sie nicht erneut explizit festgelegt werden muss. Wenn der Benutzer versucht, eine Quellidentität festzulegen, die sich von dem Wert unterscheidet, der bei der Übernahme von `CriticalRole` festgelegt wurde, wird die Anforderung der Rollenübernahme abgelehnt.

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Wenn der aufrufende Auftraggeber die Rolle übernimmt, bleibt die Quellidentität in der Anforderung ab der ersten angenommenen Rollensitzung bestehen. Daher sind sowohl der `aws:SourceIdentity`- als auch der `sts:SourceIdentity`-Schlüssel im Anfragekontext vorhanden.

Quellenidentität anzeigen in CloudTrail

Hier können Sie die Anfragen CloudTrail zur Übernahme von Rollen oder zur Zusammenführung von Benutzern anzeigen. Sie können auch die Rolle oder die Benutzeranfragen zur Durchführung von Aktionen in AWS einsehen. Die CloudTrail Protokolldatei enthält Informationen über die Quellidentität, die für die Sitzung mit übernommener Rolle oder Verbundbenutzer festgelegt wurde. Weitere Informationen finden Sie unter Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail.

Gehen Sie beispielsweise davon aus, dass ein Benutzer eine AWS STS `AssumeRole` Anfrage stellt und eine Quellidentität festlegt. Sie finden die `sourceIdentity` Informationen im `requestParameters` Schlüssel in Ihrem CloudTrail Protokoll.

Example Beispiel für einen RequestParameter-Abschnitt in einem Protokoll AWS CloudTrail

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Wenn der Benutzer die angenommene Rollensitzung verwendet, um eine Aktion auszuführen, sind die Informationen zur Quellidentität im `userIdentity` Schlüssel im CloudTrail Protokoll enthalten.

Example Beispiel für einen UserIdentity-Schlüssel in einem Protokoll AWS CloudTrail

```
{
  "eventVersion": "1.08",
  "userIdentity": {
```

```
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

Beispiele für AWS STS API-Ereignisse in CloudTrail Protokollen finden Sie unter. Beispiel für IAM-API-Ereignisse im Protokoll CloudTrail Weitere Informationen zu den in CloudTrail Protokolldateien enthaltenen Informationen finden Sie unter CloudTrail Event Reference im AWS CloudTrail Benutzerhandbuch.

Berechtigungen für GetFederationToken

Die `GetFederationToken`-Operation wird von einem IAM-Benutzer aufgerufen und gibt temporäre Anmeldedaten für diesen Benutzer zurück. Diese Operation verbindet die Benutzer. Die dem Verbundbenutzer zugewiesenen Berechtigungen sind an einem von zwei Orten definiert:

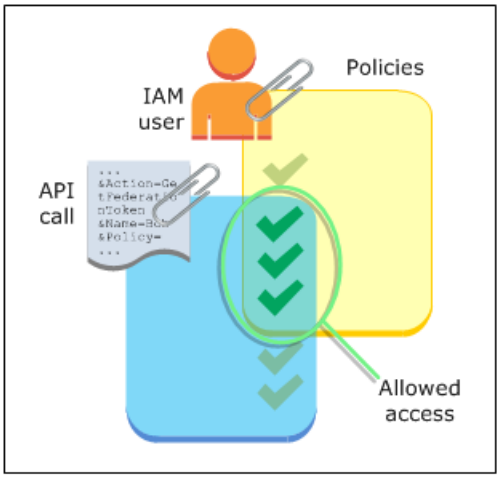
- In den Sitzungsrichtlinien, die als Parameter im API-Aufruf `GetFederationToken` übergeben werden. (Dies ist am üblichsten.)
- Eine ressourcenbasierte Richtlinie, in der der Name des Verbundbenutzers im `Principal`-Element der Richtlinie explizit aufgeführt wird. (Dies ist weniger üblich.)

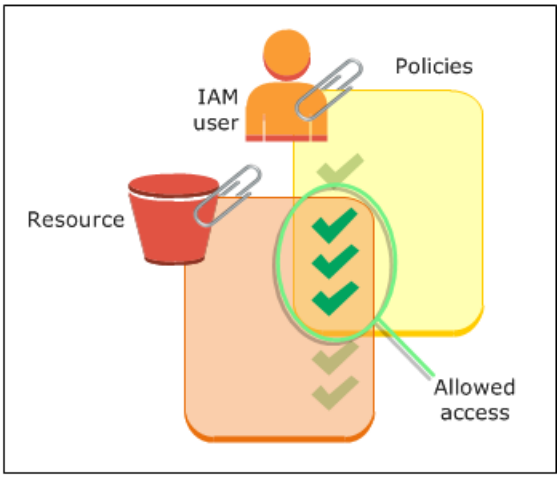
Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung programmgesteuert erstellen. Wenn Sie eine verbundene Benutzersitzung

erstellen und Sitzungsrichtlinien übergeben, sind die resultierenden Sitzungsberechtigungen eine Schnittmenge der identitätsbasierten Richtlinie des -Benutzers und der Sitzungsrichtlinien. Sie können mit der Sitzungsrichtlinie nicht mehr Berechtigungen erteilen, als durch die identitätsbasierte Richtlinie des Benutzers, der verbunden wird, zulässig sind.

Wenn Sie also keine Richtlinie mit dem `GetFederationToken`-API-Aufruf übergeben, verfügen die resultierenden temporären Sicherheitsanmeldedaten über keine Berechtigungen. Allerdings kann eine ressourcenbasierte Richtlinie zusätzliche Berechtigungen für die Sitzung bereitstellen. Sie können auf eine Ressource mit einer ressourcenbasierten Richtlinie zugreifen, die Ihre Sitzung als zulässigen Auftraggeber angibt.

Die folgenden Abbildungen zeigen eine visuelle Darstellung der Interaktion von Richtlinien, um die Berechtigungen für die von einem `GetFederationToken`-Aufruf zurückgegebenen temporären Anmeldeinformationen zu bestimmen.





Beispiel: Zuweisen von Berechtigungen mit GetFederationToken

Sie können die `GetFederationToken`-API-Aktion mit verschiedenen Arten von Richtlinien verwenden. Nachfolgend sind einige Beispiele aufgeführt.

Zum IAM-Benutzer angefügte Richtlinie

In diesem Beispiel verfügen Sie über eine browserbasierte Clientanwendung, die sich auf zwei Backend-Web-Services stützt. Ein Backend-Service ist Ihr eigener Authentifizierungsserver, der Ihr eigenes Identitätssystem zum Authentifizieren der Clientanwendung verwendet. Der andere Backend-Service ist ein AWS -Service, der einige Funktionalitäten der Clientanwendung bereitstellt. Die Clientanwendung wird von Ihrem Server authentifiziert und Ihr Server erstellt die entsprechende Berechtigungsrichtlinie oder ruft sie ab. Der Server ruft dann das `GetFederationToken`-

API auf, um temporäre Anmeldeinformationen zu erhalten und diese Anmeldeinformationen zur Clientanwendung zurückzugeben. Die Client-Anwendung kann dann mit den temporären Sicherheitsanmeldedaten Anfragen direkt an den AWS Dienst richten. Diese Architektur ermöglicht es der Client-Anwendung, AWS Anfragen zu stellen, ohne langfristige AWS Anmeldeinformationen einzubetten.

Der Authentifizierungsserver ruft die `GetFederationToken`-API mit den langfristigen Sicherheitsanmeldedaten eines IAM-Benutzers mit dem Namen `token-app` auf. Die langfristigen IAM-Benutzeranmeldedaten bleiben jedoch auf Ihrem Server und werden unter keinen Umständen an den Client weitergegeben. Im folgenden Beispiel wird die Richtlinie dem IAM-Benutzer `token-app` angefügt und die von Ihren Verbundbenutzern (Clients) benötigten Berechtigungen definiert. Beachten Sie, dass die Berechtigung `sts:GetFederationToken` für den Authentifizierungsservice erforderlich ist, damit die temporären Anmeldeinformationen für die Verbundbenutzer abgerufen werden können.

> **Note**
>
> AWS stellt zu diesem Zweck eine Java-Beispielanwendung bereit, die Sie hier herunterladen können: Token Vending Machine for Identity Registration — Beispiel für eine Java-Webanwendung.

Example Die dem IAM-Benutzer **`token-app`** angefügte Richtlinie, die **`GetFederationToken`** aufruft

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

Die obige Richtlinie erteilt dem IAM-Benutzer mehrere Berechtigungen. Doch diese Richtlinie allein erteilt dem Verbundbenutzer keinerlei Berechtigungen. Wenn dieser IAM-Benutzer `GetFederationToken` aufruft und keine Richtlinie als Parameter des API-Aufrufs übergibt, hat der resultierende Verbundbenutzer keine effektiven Berechtigungen.

Als Parameter übergebene Sitzungsrichtlinie

Die am häufigsten verwendete Methode, um sicherzustellen, dass dem Verbundbenutzer die entsprechende Berechtigung erteilt wird, besteht darin, Sitzungsrichtlinien im `GetFederationToken`-API-Aufruf zu übergeben. Aufbauend auf dem vorherigen Beispiel nehmen wir an, das `GetFederationToken` mit den Anmeldedaten des IAM-Benutzers `token-app` aufgerufen wird. Angenommen, die folgende Sitzungsrichtlinie wird als Parameter des API-Aufrufs übergeben. Der resultierende Verbundbenutzer verfügt über die Berechtigung zum Auflisten der Inhalte des Amazon S3-Buckets mit dem Namen `productionapp`. Der Benutzer kann die Amazon S3-Aktionen `GetObject`, `PutObject` und `DeleteObject` für Elemente im `productionapp`-Bucket nicht ausführen.

Dem Verbundbenutzer werden diese Berechtigungen zugewiesen, da die Berechtigungen die Schnittmenge der IAM-Benutzerberechtigungen und der Sitzungsrichtlinien bilden, die Sie übergeben.

Der Verbundbenutzer konnte keine Aktionen in Amazon SNS, Amazon SQS, Amazon DynamoDB oder in einem S3-Bucket außer `productionapp`. Diese Aktionen werden abgelehnt, auch wenn diese Berechtigungen dem IAM-Benutzer erteilt werden, der mit dem `GetFederationToken`-Aufruf verknüpft ist.

Example Als Parameter im API-Aufruf **`GetFederationToken`** übergebene Sitzungsrichtlinie

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Ressourcenbasierte Richtlinien

Einige AWS Ressourcen unterstützen ressourcenbasierte Richtlinien, und diese Richtlinien bieten einen weiteren Mechanismus, um einem Verbundbenutzer direkt Berechtigungen zu erteilen. Nur einige AWS Dienste unterstützen ressourcenbasierte Richtlinien. Amazon S3 verfügt beispielsweise über Buckets, Amazon SNS über Themen und Amazon SQS über Warteschlangen, denen Sie Richtlinien zuordnen können. Eine Liste aller Services, die ressourcenbasierte Richtlinien unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Überprüfen Sie die Spalte "Ressourcenbasierte Richtlinien" der Tabellen. Sie können ressourcenbasierte Richtlinien verwenden, um einem Verbundbenutzer Berechtigungen direkt zuzuweisen. Geben Sie hierzu den Amazon-Ressourcenname (ARN) des Verbundbenutzers `Principal`-Element der ressourcenbasierten Richtlinie an. Das folgende Beispiel veranschaulicht diesen Vorgang und ist eine Erweiterung der vorherigen Beispiele, wobei ein S3-Bucket mit dem Namen `productionapp` verwendet wird.

Die folgende ressourcenbasierte Richtlinie wird dem Bucket angefügt. Diese Bucket-Richtlinie gewährt der Verbundbenutzerin mit dem Namen Carol Zugriff auf den Bucket. Wenn die vorher beschriebene Beispielrichtlinie dem `token-app`-IAM-Benutzer angefügt worden ist, verfügt die Verbundbenutzerin mit dem Namen Carol über die Berechtigung zum Ausführen der Aktionen `s3:GetObject`, `s3:PutObject`, und `s3:DeleteObject` im Bucket mit dem

Namen `productionapp`. Dies gilt auch, wenn keine Sitzungsrichtlinie als Parameter im API-Aufruf `GetFederationToken` übergeben wird. Dies liegt daran, dass in diesem Fall der Verbundbenutzerin mit dem Namen Carol durch die folgende ressourcenbasierte Richtlinie explizit Berechtigungen erteilt worden sind.

Beachten Sie, dass ein Verbundbenutzer nur dann Berechtigungen erhält, wenn diese Berechtigungen sowohl dem IAM-Benutzer als auch dem Verbundbenutzer erteilt werden. Sie können auch (innerhalb des Kontos) durch eine ressourcenbasierte Richtlinie gewährt werden, die den Verbundbenutzer im `Principal`-Element der Richtlinie ausdrücklich benennt, wie im folgenden Beispiel.

Example Bucket-Richtlinie, die dem Verbundbenutzer Zugriffsberechtigungen erteilt

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "arn:aws:sts::account-id:federated-user/Carol"},
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::productionapp/*"]
  }
}
```

Weitere Informationen über das Auswerten von Richtlinien finden Sie unter Logik der Richtlinienevaluierung.

Berechtigungen für GetSessionToken

Primärer Anlass zum Aufruf der API-Operation `GetSessionToken` oder des CLI-Befehls `get-session-token` ist, wenn ein Benutzer mit Multi-Factor Authentication (MFA) authentifiziert werden muss. Es ist möglich, eine Richtlinie zu verfassen, die bestimmte Aktionen nur dann erlaubt, wenn diese Aktionen von einem Benutzer angefordert werden, der mit MFA authentifiziert wurde. Um die MFA-Autorisierungsprüfung erfolgreich zu bestehen, muss ein Benutzer zunächst `GetSessionToken` aufrufen und die optionalen Parameter `SerialNumber` und `TokenCode` einschließen. Wenn sich der Benutzer erfolgreich bei einem MFA-Gerät authentifiziert, enthalten die von der API-Operation `GetSessionToken` zurückgegebenen Anmeldeinformationen den MFA-

Kontext. Dieser Kontext gibt an, dass der Benutzer mit MFA authentifiziert und für API-Operationen autorisiert ist, für die eine MFA-Authentifizierung erforderlich ist.

Berechtigungen erforderlich für GetSessionToken

Ein Benutzer benötigt keine Berechtigungen, um ein Sitzungstoken zu erhalten. Der Zweck der Operation `GetSessionToken` besteht darin, den Benutzer mithilfe von MFA zu authentifizieren. Richtlinien können nicht dazu verwendet werden, Authentifizierungsoperationen zu steuern.

Um Berechtigungen für die Ausführung der meisten AWS Operationen zu erteilen, fügen Sie die Aktion mit dem gleichen Namen zu einer Richtlinie hinzu. Um einen Benutzer zu erstellen, müssen Sie z. B. die API-Operation `CreateUser`, den CLI-Befehl `create-user` oder die AWS Management Console verwenden. Um diese Operationen durchführen zu können, müssen Sie über eine Richtlinie verfügen, die Ihnen Zugriff auf die Aktion `CreateUser` gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateUser",
            "Resource": "*"
        }
    ]
}
```

Sie können die Aktion `GetSessionToken` in Ihre Richtlinien einfügen, dies hat aber keine Auswirkungen auf die Fähigkeit des Benutzers, die Operation `GetSessionToken` auszuführen.

Berechtigungen von GetSessionToken

Wenn `GetSessionToken` mit den Anmeldeinformationen eines IAM-Benutzers aufgerufen wird, haben die temporären Sicherheitsanmeldeinformationen dieselben Berechtigungen wie der IAM-Benutzer. In ähnlicher Weise verfügen die temporären Root-Benutzer des AWS-Kontos Sicherheitsanmeldedaten über Root-Benutzerberechtigungen, wenn `GetSessionToken` sie mit Anmeldeinformationen aufgerufen wird.

> **Note**
>
> Es wird empfohlen, `GetSessionToken` nicht mit Stammbenutzer-Anmeldeinformationen aufzurufen. Befolgen Sie stattdessen unsere bewährten Methoden und erstellen Sie IAM-

Benutzer mit den Berechtigungen, die sie benötigen. Verwenden Sie diese IAM-Benutzer dann für die tägliche Interaktion mit AWS.

Die temporären Anmeldeinformationen, die Sie erhalten, wenn Sie `GetSessionToken` aufrufen, haben die folgenden Funktionen und Einschränkungen:

- Sie können die Anmeldeinformationen für den Zugriff auf verwenden, AWS Management Console indem Sie die Anmeldeinformationen an den Single Sign-On-Endpunkt des Verbunds unter übergeben. `https://signin.aws.amazon.com/federation` Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.
- Sie können die Anmeldeinformationen nicht verwenden, um IAM- oder AWS STS -API-Operationen aufzurufen. Sie können sie verwenden, um API-Operationen für andere AWS Dienste aufzurufen.

Unter Vergleich der AWS STS API-Operationen können Sie diese API-Operation und ihre Grenzen und Funktionen mit den anderen API-Operationen, die temporäre Sicherheitsanmeldeinformationen erstellen, vergleichen.

Weitere Informationen über MFA-geschützten API-Zugriff mithilfe von `GetSessionToken` finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs.

Deaktivieren von Berechtigungen für temporäre Sicherheitsanmeldeinformationen

Temporäre Sicherheitsanmeldeinformationen sind bis zu ihrem Ablauf gültig. Diese Anmeldeinformationen sind für die angegebene Dauer von 900 Sekunden (15 Minuten) bis maximal 129 600 Sekunden (36 Stunden) gültig. Die Standardsitzungsdauer beträgt 43 200 Sekunden (12 Stunden). Sie können diese Anmeldeinformationen widerrufen, müssen aber auch die Berechtigungen für die Rolle ändern, um die Nutzung kompromittierter Anmeldeinformationen für bösartige Kontoaktivitäten zu verhindern. Berechtigungen, die temporären Sicherheitsanmeldedaten zugewiesen wurden, werden jedes Mal bewertet, wenn sie für eine AWS Anfrage verwendet werden. Sobald Sie alle Berechtigungen aus den Anmeldeinformationen entfernt haben, schlagen AWS Anfragen, die sie verwenden, fehl.

Es kann einige Minuten dauern, bis Richtlinienaktualisierungen wirksam werden. Widerrufen Sie die temporären Sicherheitsanmeldeinformationen der Rolle, um alle Benutzer, die die Rolle annehmen, zu einer erneuten Authentifizierung und der Anforderung neuer Anmeldeinformationen zu zwingen.

Sie können die Berechtigungen für eine nicht ändern Root-Benutzer des AWS-Kontos. Ebenso können die Berechtigungen für die temporären Sicherheitsanmeldeinformationen nicht geändert werden, die durch Aufrufen von `GetFederationToken` oder `GetSessionToken` erstellt wurden, während der Benutzer als Stammbenutzer angemeldet war. Daher empfehlen wir, dass Sie `GetFederationToken` oder `GetSessionToken` nicht als Stammbenutzer aufrufen.

> **Important**
>
> Sie können in IAM keine Rollen bearbeiten, die anhand von IAM Identity Center-Berechtigungssätzen erstellt wurden. Sie müssen die aktive Sitzung mit dem Berechtigungssatz für einen Benutzer in IAM Identity Center widerrufen. Weitere Informationen finden Sie unter Widerrufen aktiver IAM-Rollensitzungen, die mit Berechtigungssätzen erstellt wurden, im IAM Identity Center-Benutzerhandbuch.

Themen

- Zugriff auf alle mit einer Rolle verknüpften Sitzungen verweigern
- Zugriff auf eine bestimmte Sitzung verweigern
- Eine Benutzersitzung mit Bedingungskontextschlüsseln verweigern
- Einen Sitzungsbenutzer mit ressourcenbasierten Richtlinien verweigern

Zugriff auf alle mit einer Rolle verknüpften Sitzungen verweigern

Verwenden Sie diesen Ansatz, wenn Sie Bedenken hinsichtlich verdächtiger Zugriffe durch Folgendes haben:

- Prinzipale von einem anderen Konto mit kontoübergreifendem Zugriff
- Externe Benutzeridentitäten mit Berechtigungen für den Zugriff auf AWS Ressourcen in Ihrem Konto
- Benutzer, die in einer Mobil- oder Webanwendung bei einem OIDC-Anbieter authentifiziert wurden

Dieses Verfahren verweigert allen Benutzern, die die Berechtigung haben, eine Rolle zu übernehmen, die Berechtigung.

Um die Berechtigungen zu ändern oder zu entfernen, die den temporären Sicherheitsanmeldeinformationen zugewiesen sind, die Sie durch Aufrufen von `AssumeRole`,

`AssumeRoleWithSAML` oder `AssumeRoleWithWebIdentity`, `GetFederationToken` oder `GetSessionToken` erhalten, können Sie die Berechtigungsrichtlinie, die die Berechtigungen für die Rolle definiert, bearbeiten oder löschen.

> **Important**
>
> Wenn es eine ressourcenbasierte Richtlinie gibt, die dem Prinzipal Zugriff gewährt, müssen Sie auch eine explizite Ablehnung für diese Ressource hinzufügen. Details dazu finden Sie unter Einen Sitzungsbenutzer mit ressourcenbasierten Richtlinien verweigern.

1. Melden Sie sich bei der IAM-Konsole an AWS Management Console und öffnen Sie sie.
2. Wählen Sie im Navigationsbereich den Namen der Rolle aus, die Sie bearbeiten möchten. Sie können das Suchfeld verwenden, um die Liste zu filtern.
3. Wählen Sie die entsprechende Richtlinie aus.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wählen Sie die Registerkarte JSON und aktualisieren Sie die Richtlinie, um alle Ressourcen und Aktionen zu verweigern.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

6. Überprüfen Sie auf der Seite Review (Prüfen) unter Summary (Übersicht) die Richtlinienzusammenfassung und wählen Sie dann Save changes (Änderungen speichern) aus, um Ihre Eingaben zu speichern.

Wenn Sie die Richtlinie aktualisieren, wirken sich die Änderungen auf die Berechtigungen aller temporären Anmeldeinformationen aus, die mit der Rolle verknüpft sind. Dies gilt auch für Anmeldeinformationen, die ausgestellt wurden, bevor Sie die Richtlinie für die Berechtigungen der Rolle geändert haben. Nachdem Sie die Richtlinie aktualisiert haben, können Sie die temporären

Sicherheitsanmeldeinformationen der Rolle widerrufen, um sofort alle Berechtigungen für die ausgegebenen Anmeldeinformationen der Rolle zu entziehen.

Zugriff auf eine bestimmte Sitzung verweigern

Wenn Sie die Rollen, die von einem IdP übernommen werden können, mit einer „Alles verweigern“-Richtlinie aktualisieren oder die Rolle vollständig löschen, werden alle Benutzer, die Zugriff auf die Rolle haben, beeinträchtigt. Sie können den Zugriff basierend auf dem `Principal`-Element verweigern, ohne dass dies Auswirkungen auf die Berechtigungen aller anderen Sitzungen, die mit der Rolle verknüpft sind, hat.

Dem `Principal` können mithilfe von Bedingungskontextschlüsseln oder ressourcenbasierten Richtlinien Berechtigungen verweigert werden.

> **Tip**
>
> Sie können die ARNs von Verbundbenutzern mithilfe von Protokollen ermitteln. AWS CloudTrail Weitere Informationen finden Sie unter So identifizieren Sie Ihre Verbundbenutzer auf einfache Weise mithilfe von. AWS CloudTrail

Eine Benutzersitzung mit Bedingungskontextschlüsseln verweigern

Sie können Bedingungskontextschlüssel in Situationen verwenden, in denen Sie den Zugriff auf bestimmte temporäre Anmeldeinformationssitzungen verweigern möchten, ohne dass sich dies auf die Berechtigungen des IAM-Benutzers oder der Rolle auswirkt, der/die die Anmeldeinformationen erstellt hat.

Weitere Informationen zu Bedingungskontextschlüsseln finden Sie unter AWS Kontextschlüssel für globale Bedingungen.

> **Note**
>
> Wenn es eine ressourcenbasierte Richtlinie gibt, die dem Prinzipal Zugriff gewährt, müssen Sie nach Abschluss dieser Schritte auch eine explizite Verweigerungsanweisung zur ressourcenbasierten Richtlinie hinzufügen.

Nachdem Sie die Richtlinie aktualisiert haben, können Sie die temporären Anmeldeinformationen der Rolle widerrufen, um alle ausgestellten Anmeldeinformationen sofort zu entziehen.

als: PrincipalArn

Sie können den Bedingungskontextschlüssel war: PrincipalArn verwenden, um den Zugriff auf einen bestimmten Prinzipal-ARN zu verweigern. Dazu geben Sie im Bedingungselement einer Richtlinie die eindeutige Kennung (ID) des IAM-Benutzers, der Rolle oder des Verbundbenutzers an, dem die temporären Anmeldeinformationen zugeordnet sind.

1. Wählen Sie im Navigationsbereich der IAM-Konsole den Namen der Rolle aus, die Sie bearbeiten möchten. Sie können das Suchfeld verwenden, um die Liste zu filtern.
2. Wählen Sie die entsprechende Richtlinie aus.
3. Wählen Sie die Registerkarte Berechtigungen.
4. Wählen Sie die Registerkarte JSON und fügen Sie eine Verweigerungsanweisung für den Prinzipal-ARN hinzu, wie im folgenden Beispiel gezeigt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

5. Überprüfen Sie auf der Seite Review (Prüfen) unter Summary (Übersicht) die Richtlinienzusammenfassung und wählen Sie dann Save changes (Änderungen speichern) aus, um Ihre Eingaben zu speichern.

aws:userid

Mit dem Bedingungskontextschlüssel aws:userid können Sie den Zugriff auf alle oder bestimmte temporäre Sicherheitsanmeldeinformationssitzungen verweigern, die mit dem IAM-Benutzer oder der IAM-Rolle verknüpft sind. Dazu geben Sie im `Condition`-Element einer Richtlinie die eindeutige Kennung (ID) des IAM-Benutzers, der IAM-Rolle oder des Verbundbenutzers an, mit dem die temporären Sicherheitsanmeldeinformationen verknüpft sind.

Die folgende Richtlinie zeigt ein Beispiel dafür, wie Sie den Zugriff auf temporäre Sicherheitsanmeldeinformationssitzungen mithilfe des Bedingungskontextschlüssels `aws:userid` verweigern können.

- `AIDAXUSER1` stellt die eindeutige Kennung für einen IAM-Benutzer dar. Durch die Angabe der eindeutigen Kennung eines IAM-Benutzers als Wert für den Kontextschlüssel `aws:userid` werden alle mit dem IAM-Benutzer verknüpften Sitzungen abgelehnt.
- `AROAXROLE1` stellt die eindeutige Kennung für eine IAM-Rolle dar. Wenn Sie den eindeutigen Bezeichner einer IAM-Rolle als Wert für den Kontextschlüssel `aws:userid` angeben, werden alle mit der Rolle verknüpften Sitzungen abgelehnt.
- `AROAXROLE2` stellt die eindeutige Kennung für eine Sitzung mit angenommener Rolle dar. Im Abschnitt „ caller-specified-role-session-name“ des eindeutigen Bezeichners für die angenommene Rolle können Sie einen Namen für die Rollensitzung oder ein Platzhalterzeichen angeben, wenn der StringLike Bedingungsoperator verwendet wird. Wenn Sie den Namen der Rollensitzung angeben, wird die benannte Rollensitzung verweigert, ohne dass dies Auswirkungen auf die Berechtigungen der Rolle hat, die die Anmeldeinformationen erstellt hat. Durch die Angabe eines Platzhalters für den Rollensitzungsnamen, werden alle mit der Rolle verknüpften Sitzungen abgelehnt.
- `account-id:<federated-user-caller-specified-name>` stellt die eindeutige Kennung für eine Verbundbenutzersitzung dar. Ein Verbundbenutzer wird von einem IAM-Benutzer erstellt, der die API aufruft. GetFederationToken Wenn Sie die eindeutige Kennung für einen Verbundbenutzer angeben, wird die Sitzung des benannten Verbundbenutzers verweigert, ohne dass dies Auswirkungen auf die Berechtigungen der Rolle hat, die die Anmeldeinformationen erstellt hat.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "account-id:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Konkrete Beispiele für Hauptschlüsselwerte finden Sie unter Auftraggeber-Schlüsselwerte. Informationen zu eindeutigen IAM-Identifikatoren finden Sie unter Eindeutige Bezeichner.

Einen Sitzungsbenutzer mit ressourcenbasierten Richtlinien verweigern

Wenn der Prinzipl-ARN auch in ressourcenbasierten Richtlinien enthalten ist, müssen Sie den Zugriff auch basierend auf den `principalId`- oder `sourceIdentity`-Werten des jeweiligen Benutzers im `Principal`-Element einer ressourcenbasierten Richtlinie widerrufen. Wenn Sie nur die Berechtigungsrichtlinie für die Rolle aktualisieren, kann der Benutzer weiterhin die in der ressourcenbasierten Richtlinie zulässigen Aktionen ausführen.

1. Lesen Sie in AWS Dienste, die mit IAM funktionieren nach, ob der Service ressourcenbasierte Richtlinien unterstützt.
2. Melden Sie sich bei dem an AWS Management Console und öffnen Sie die Konsole für den Dienst. Jeder Service verfügt über einen anderen Standort in der Konsole zum Anfügen von Richtlinien.
3. Bearbeiten Sie die Richtlinienanweisung, um die identifizierenden Informationen der Anmeldeinformationen anzugeben:

    a. Geben Sie unter `Principal` die ARN der Anmeldeinformation ein, die verweigert werden soll.

    b. Geben Sie unter `Effect` „Verweigern“ ein.

c. Geben Sie unter `Action` den Service-Namespace und den Namen der abzulehnenden Aktion ein. Um alle Aktionen zu verweigern, verwenden Sie das Platzhalterzeichen (*). Zum Beispiel: „s3:*.“

d. Geben Sie unter `Resource` den ARN der Zielressource ein. Zum Beispiel: „arn:aws:s3:::EXAMPLE-BUCKET.“

```
{
"Version": "2012-10-17",
  "Statement": {
    "Principal": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
    ],
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::EXAMPLE-BUCKET"
  }
}
```

4. Speichern Sie Ihre Arbeit.

Erteilen von Berechtigungen zum Erstellen von temporären Sicherheitsanmeldeinformationen

Standardmäßig haben IAM-Benutzer keine Berechtigung zum Erstellen temporärer Sicherheitsanmeldeinformationen für Verbundbenutzer und Rollen. Sie müssen eine Richtlinie verwenden, um Ihren Benutzern diese Berechtigungen zu gewähren. Obwohl Sie einem Benutzer Berechtigungen direkt erteilten können, empfehlen wir dringend, die Berechtigungen einer Gruppe zu erteilen. Dadurch ist die Verwaltung der Berechtigungen wesentlich einfacher. Wenn ein Benutzer nicht mehr die berechtigten Aufgaben ausführen muss, entfernen Sie ihn einfach aus der Gruppe. Wenn diese Aufgabe von einem anderen Benutzer auszuführen ist, fügen Sie ihn der Gruppe hinzu, um die Berechtigungen zu erteilen.

Um einer IAM-Gruppe Berechtigung zum Erstellen von temporären Sicherheitsanmeldeinformationen für Verbundbenutzer oder Rollen zu erteilen, fügen Sie eine Richtlinie an, die eine oder beide der folgenden Berechtigungen gewährt:

- Um den Verbundbenutzern den Zugriff auf eine IAM-Rolle zu ermöglichen, erteilen Sie die Zugriffsberechtigung auf AWS STS `AssumeRole` .
- Gewähren Sie Verbundbenutzern, die keine Rolle benötigen, Zugriff auf. AWS STS `GetFederationToken`

Weitere Informationen zu den Unterschieden zwischen den API-Operationen `AssumeRole` und `GetFederationToken` finden Sie unter Anfordern temporärer Sicherheitsanmeldeinformationen.

IAM-Benutzer können auch `GetSessionToken` aufrufen, um temporäre Sicherheitsanmeldeinformationen zu erstellen. Für den Aufruf von `GetSessionToken` benötigt ein Benutzer keine Berechtigungen. Der Zweck dieser Operation besteht darin, den Benutzer mithilfe von MFA zu authentifizieren. Die Authentifizierung kann nicht über Richtlinien gesteuert werden. Dies bedeutet, dass Sie IAM-Benutzer nicht daran hindern können, zum Erstellen von temporären Anmeldeinformationen `GetSessionToken` aufzurufen.

Example Beispiel für eine Richtlinie, die die Erlaubnis zur Übernahme einer Rolle erteilt

Die folgende Beispielrichtlinie gewährt die Erlaubnis, die `UpdateApp` Rolle in AWS-Konto `123123123123` aufzurufen`AssumeRole`. Wenn `AssumeRole` verwendet wird, kann der Benutzer (oder die Anwendung), der die Sicherheitsanmeldeinformationen im Namen eines Verbundbenutzers erstellt, nur die explizit in der Berechtigungsrichtlinie der Rolle angegebenen Berechtigungen delegieren.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Beispielrichtlinie, die die Erlaubnis zum Erstellen temporärer Sicherheitsnachweise für einen Verbundbenutzer erteilt

Die folgende Beispielrichtlinie zeigt, wie Sie die Zugriffsberechtigung für `GetFederationToken` erteilen.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Wenn Sie IAM-Benutzern die Berechtigung zum Erstellen von temporären Sicherheitsanmeldeinformationen für Verbundbenutzer mit `GetFederationToken` erteilen, sollten Sie sich darüber im Klaren sein, dass hiermit diesen Benutzern die Delegierung ihrer eigenen Berechtigungen ermöglicht wird. Weitere Informationen zum Delegieren von Berechtigungen zwischen IAM-Benutzern und finden Sie AWS-Konten unter. Beispiele für Richtlinien zum Delegieren des Zugriffs Weitere Informationen zum Steuern von Berechtigungen in temporären Sicherheitsanmeldeinformationen finden Sie unter Steuern von Berechtigungen für temporäre Sicherheitsanmeldeinformationen.

Example Beispielrichtlinie, die einem Benutzer die eingeschränkte Berechtigung erteilt, temporäre Sicherheitsnachweise für Benutzer im Verbund zu erstellen

Wenn Sie einem IAM-Benutzer ermöglichen, `GetFederationToken` aufzurufen, ist es eine bewährte Methode, die Berechtigungen einzuschränken, die der IAM-Benutzer delegieren kann. Folgende Richtlinie zeigt beispielsweise, wie einem IAM-Benutzer ermöglicht wird, temporäre Sicherheitsanmeldeinformationen nur für Verbundbenutzer zu erstellen, deren Namen mit Manager beginnt.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Erteilen von Berechtigungen zur Verwendung identitätsbewusster Konsolensitzungen

Mithilfe identitätsbewusster Konsolensitzungen können AWS IAM Identity Center Benutzer- und Sitzungs-IDs in die AWS Konsolensitzungen der Benutzer aufgenommen werden, wenn sie sich anmelden. Amazon Q Developer ProVerwendet beispielsweise identitätsbewusste Konsolensitzungen, um das Serviceerlebnis zu personalisieren. Weitere Informationen zu identitätsbewussten Konsolensitzungen finden Sie im Benutzerhandbuch unter Aktivieren identitätsbewusster Konsolensitzungen.AWS IAM Identity Center Informationen zur Einrichtung von Amazon Q Developer finden Sie unter Einrichten von Amazon Q Developer im Amazon Q Developer User Guide.

Damit einem Benutzer identitätsbewusste Konsolensitzungen zur Verfügung stehen, müssen Sie eine identitätsbasierte Richtlinie verwenden, um dem IAM-Prinzipal die `sts:SetContext` Berechtigung für die Ressource zu erteilen, die seine eigene Konsolensitzung darstellt.

> **Important**
>
> Standardmäßig sind Benutzer nicht berechtigt, den Kontext für ihre identitätsbewussten Konsolensitzungen festzulegen. Um dies zu ermöglichen, müssen Sie dem IAM-Prinzipal die `sts:SetContext` entsprechende Berechtigung in einer identitätsbasierten Richtlinie erteilen, wie im folgenden Richtlinienbeispiel dargestellt.

Das folgende Beispiel für eine identitätsbasierte Richtlinie erteilt einem IAM-Prinzipal die `sts:SetContext` Berechtigung, sodass der Prinzipal den identitätsbewussten Konsolensitzungskontext für seine eigenen Konsolensitzungen festlegen kann. AWS Die Richtlinienressource,, stellt die Sitzung des `arn:aws:sts::`*`account-id`*`:self` Aufrufers dar. AWS Das `account-id` ARN-Segment kann `*` in Fällen, in denen dieselbe Berechtigungsrichtlinie für mehrere Konten bereitgestellt wird, durch ein Platzhalterzeichen ersetzt werden, z. B. wenn diese Richtlinie mithilfe von IAM Identity Center-Berechtigungssätzen bereitgestellt wird.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:SetContext",
      "Resource": "arn:aws:sts::account-id:self"
    }
```

```
  ]
}
```

Verwaltung AWS STS in einem AWS-Region

Standardmäßig ist AWS Security Token Service (AWS STS) als globaler Dienst verfügbar, und alle AWS STS Anfragen gehen an einen einzigen Endpunkt unter`https://sts.amazonaws.com`. AWS empfiehlt die Verwendung regionaler AWS STS Endpunkte anstelle des globalen Endpunkts, um die Latenz zu reduzieren, Redundanz zu gewährleisten und die Gültigkeit von Sitzungstoken zu erhöhen.

- Latenz reduzieren — Indem Sie Ihre AWS STS Anrufe an einen Endpunkt tätigen, der geografisch näher an Ihren Diensten und Anwendungen liegt, können Sie auf AWS STS Dienste mit geringerer Latenz und besseren Reaktionszeiten zugreifen.
- Integrierte Redundanz – Sie können die Auswirkungen eines Fehlers innerhalb eines Workloads auf eine begrenzte Anzahl von Komponenten mit einem vorhersehbaren Umfang der Auswirkungen einschränken. Durch die Verwendung regionaler AWS STS Endpunkte können Sie den Umfang Ihrer Komponenten an den Umfang Ihrer Sitzungstoken anpassen. Weitere Informationen zu dieser Zuverlässigkeitssäule finden Sie unter Verwendung der Fehlerisolierung zum Schutz Ihres Workloads im AWS -Well-Architected-Framework.
- Erhöhen Sie die Gültigkeit von Sitzungstoken — Sitzungstoken von regionalen AWS STS Endpunkten sind insgesamt gültig. AWS-Regionen Sitzungstoken vom globalen STS-Endpunkt sind nur gültig AWS-Regionen , wenn sie standardmäßig aktiviert sind. Wenn Sie beabsichtigen, eine neue Region für Ihr Konto zu aktivieren, können Sie Sitzungstoken von regionalen AWS STS Endpunkten verwenden. Wenn Sie den globalen Endpunkt verwenden möchten, müssen Sie die Regionskompatibilität der AWS STS Sitzungstoken für den globalen Endpunkt ändern. Dadurch wird sichergestellt, dass alle Token gültig sind AWS-Regionen.

Verwalten von Sitzungstoken des globalen Endpunkts

Die meisten AWS-Regionen sind AWS-Services standardmäßig für alle Operationen aktiviert. Diese Regionen werden automatisch für die Verwendung mit aktiviert AWS STS. Einige Regionen, wie z. B. Asien-Pazifik (Hongkong), müssen manuell aktiviert werden. Weitere Informationen zur Aktivierung und Deaktivierung finden Sie im AWS Account Management Referenzhandbuch unter Geben Sie an AWS-Regionen, welche AWS-Regionen Konten verwendet werden können. Wenn Sie diese AWS Regionen aktivieren, werden sie automatisch für die Verwendung mit AWS STS aktiviert. Sie können den AWS STS Endpunkt nicht für eine Region aktivieren, die deaktiviert ist. Sitzungstoken, die in allen Fällen gültig sind, AWS-Regionen enthalten mehr Zeichen als Token, die in Regionen gültig

sind, die standardmäßig aktiviert sind. Das Ändern dieser Einstellung kann sich auf vorhandene Systeme auswirken, in denen Sie Token vorübergehend speichern.

Sie können diese Einstellung mithilfe der AWS API AWS Management Console AWS CLI, oder ändern.

So ändern Sie die Vereinbarkeit der Region für die Sitzungstoken des globalen Endpunkts (Konsole)

1. Melden Sie sich als Root-Benutzer oder als IAM-Benutzer mit der Berechtigung zur Durchführung von IAM-Verwaltungsaufgaben an. Um die Vereinbarkeit der Sitzungstoken zu ändern, benötigen Sie eine Richtlinie, die die `iam:SetSecurityTokenServicePreferences`-Aktion zulässt.
2. Öffnen Sie die IAM-Konsole. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).
3. Im Abschnitt Security Token Service (STS) finden Sie Sitzungstoken von den STS-Endpunkten. Der globale Endpunkt gibt `Valid only in AWS-Regionen enabled by default` an. Wählen Sie Change.
4. Wählen Sie im Dialogfeld „Regionskompatibilität ändern“ die Option Alle aus AWS-Regionen. Wählen Sie dann Save changes (Änderungen speichern).

> **Note**
>
> Sitzungstoken, die in allen Fällen gültig sind, AWS-Region enthalten mehr Zeichen als Token, die in Regionen gültig sind, die standardmäßig aktiviert sind. Das Ändern dieser Einstellung kann sich auf vorhandene Systeme auswirken, in denen Sie Token vorübergehend speichern.

So ändern Sie die Vereinbarkeit der Region für die Sitzungstoken des globalen Endpunkts (AWS CLI)

Legen Sie die Version des Sitzungs-Tokens fest. Token der Version 1 sind nur gültig AWS-Regionen , wenn sie standardmäßig verfügbar sind. Diese Token funktionieren icht in manuell aktivierten Regionen, wie z. B. Asien-Pazifik (Hongkong), nicht. Token der Version 2 sind in allen Regionen gültig. Token der Version 2 enthalten jedoch mehr Zeichen und können sich auf Systeme auswirken, in denen Sie Token vorübergehend speichern.

- `aws iam set-security-token-service-preferences`

So ändern Sie die Vereinbarkeit der Region für die Sitzungstoken des globalen Endpunkts (AWS - API)

Legen Sie die Version des Sitzungs-Tokens fest. Token der Version 1 sind nur gültig AWS-Regionen , wenn sie standardmäßig verfügbar sind. Diese Token funktionieren icht in manuell aktivierten Regionen, wie z. B. Asien-Pazifik (Hongkong), nicht. Token der Version 2 sind in allen Regionen gültig. Token der Version 2 enthalten jedoch mehr Zeichen und können sich auf Systeme auswirken, in denen Sie Token vorübergehend speichern.

- `SetSecurityTokenServicePreferences`

Aktivierung und Deaktivierung in einem AWS STSAWS-Region

Wenn Sie STS-Endpunkte für eine Region aktivieren, AWS STS kann temporäre Anmeldeinformationen für Benutzer und Rollen in Ihrem Konto ausgestellt werden, die eine AWS STS Anfrage stellen. Diese Anmeldedaten können dann in einer beliebigen Region verwendet werden, die standardmäßig oder manuell aktiviert ist. Für Regionen, die standardmäßig aktiviert sind, müssen Sie den regionalen STS-Endpunkt in dem Konto aktivieren, in dem die temporären Anmeldeinformationen generiert werden. Es spielt beim Senden der Anforderung keine Rolle, ob ein Benutzer bei diesem oder einem anderen Konto angemeldet ist. Bei Regionen, die manuell aktiviert werden, müssen Sie die Region sowohl in dem Konto, das die Anforderung stellt, als auch in dem Konto, in dem die temporären Anmeldeinformationen generiert werden, aktivieren.

Stellen Sie sich zum Beispiel vor, ein Benutzer in Konto A möchte eine `sts:AssumeRole` API-Anfrage an den AWS STS regionalen Endpunkt `https://sts.ap-east-1.amazonaws.com` senden. Die Anforderung gilt für die temporären Anmeldeinformationen für die Rolle `Developer` in Konto B. Da die Anforderung zum Erstellen von Anmeldeinformationen für eine Entität des Kontos B gesendet wird, muss Konto B die Region `ap-east-1` aktivieren. Benutzer aus Konto A (oder einem anderen Konto) können den Endpunkt `ap-east-1` AWS STS aufrufen, um Anmeldeinformationen für Konto B anzufordern. Dabei spielt es keine Rolle, ob die Region in ihren Konten aktiviert ist oder nicht.

> **Note**
>
> Aktive Regionen stehen allen Benutzern zur Verfügung, die temporäre Anmeldedaten in diesem Konto verwendet. Um zu steuern, welche IAM-Benutzer oder Rollen auf die Region

zugreifen können, verwenden Sie den Bedingungsschlüssel `aws:RequestedRegion` in Ihren Berechtigungsrichtlinien.

Zur Aktivierung oder Deaktivierung AWS STS in einer Region, die standardmäßig aktiviert ist (Konsole)

1. Melden Sie sich als Root-Benutzer oder als IAM-Benutzer mit der Berechtigung zur Durchführung von IAM-Verwaltungsaufgaben an.
2. Öffnen Sie die IAM-Konsole und wählen Sie im Navigationsbereich Kontoeinstellungen.
3. Suchen Sie im Abschnitt Security Token Service (STS) Endpoints (Endpunkte) nach der Region, die Sie konfigurieren möchten, und wählen Sie dann in der Spalte STS status (STS-Status) Active (Aktiv) oder Inactive (Inaktiv) aus.
4. Wählen Sie in dem sich öffnenden Dialogfeld Activate (Aktivieren) oder Deactivate (Deaktivieren).

Regionen, die aktiviert werden müssen, werden AWS STS automatisch aktiviert, wenn Sie die Region aktivieren. Nachdem Sie eine Region aktiviert haben, AWS STS ist sie immer für die Region aktiv und kann nicht deaktiviert werden. Weitere Informationen zur Aktivierung von Regionen, die standardmäßig deaktiviert sind, finden Sie im AWS Account Management Referenzhandbuch unter Spezifizieren, welche Regionen für AWS-Regionen Ihr Konto verwendet werden können.

Schreiben von Code zum Verwenden von AWS STS -Regionen

Nachdem Sie eine Region aktiviert haben, können Sie AWS STS API-Aufrufe an diese Region weiterleiten. Der folgende Java-Codeausschnitt zeigt, wie Sie ein `AWSSecurityTokenService` Objekt so konfigurieren, dass es Anfragen an die Region Europa (Mailand) (eu-south-1) sendet.

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS empfiehlt, dass Sie Aufrufe an einen regionalen Endpunkt tätigen. Informationen zum manuellen Aktivieren einer Region finden Sie im AWS Account Management Referenzhandbuch unter Geben Sie an, welche Region für AWS-Regionen Ihr Konto verwendet werden kann.

In diesem Beispiel instanziiert die erste Zeile ein `EndpointConfiguration`-Objekt namens `regionEndpointConfig`, wobei die URL des Endpunkts und die AWS-Region als Parameter übergeben werden.

Informationen zum Einrichten AWS STS regionaler Endpunkte mithilfe einer Umgebungsvariablen für AWS SDKs finden Sie unter AWS STS Regionalisierte Endpunkte im Referenzhandbuch für AWS SDKs und Tools.

Angaben zu allen anderen Sprach- und Programmierumgebungskombinationen finden Sie in der Dokumentation des entsprechenden SDK.

Regionen und Endpunkte

In der folgenden Tabelle sind die Regionen und deren Endpunkte aufgelistet. Es wird angegeben, welche standardmäßig aktiviert sind und welche Sie selbst aktivieren oder deaktivieren können.

Name der Region	Endpunkt	Standardmäßig aktiviert	Manuelles Aktivieren/ Deaktivieren
--Global--	sts.amazonaws.com	 Ja	 Nein
USA Ost (Ohio)	sts.us-east-2.amazonaws.com	 Ja	 Ja
USA Ost (Nord-Virginia)	sts.us-east-1.amazonaws.com	 Ja	 Nein

Name der Region	Endpunkt	Standardmäßig aktiviert	Manuelles Aktivieren/ Deaktivieren
USA West (Nordkalifornien)	sts.us-west-1.amazonaws.com	Ja	Ja
USA West (Oregon)	sts.us-west-2.amazonaws.com	Ja	Ja
Afrika (Kapstadt)	sts.af-south-1.amazonaws.com	Nein[1]	Nein
Asien-Pazifik (Hongkong)	sts.ap-east-1.amazonaws.com	Nein[1]	Nein
Asien-Pazifik (Hyderabad)	sts.ap-south-2.amazonaws.com	Nein[1]	Nein
Asien-Pazifik (Jakarta)	sts.ap-southeast-3.amazonaws.com	Nein[1]	Nein

Name der Region	Endpunkt	Standardmäßig aktiviert	Manuelles Aktivieren/ Deaktivieren
Asien-Pazifik (Melbourne)	sts.ap-southeast-4.amazonaws.com	Nein[1]	Nein
Asien-Pazifik (Mumbai)	sts.ap-south-1.amazonaws.com	Ja	Ja
Asien-Pazifik (Osaka)	sts.ap-northeast-3.amazonaws.com	Ja	Ja
Asien-Pazifik (Seoul)	sts.ap-northeast-2.amazonaws.com	Ja	Ja
Asien-Pazifik (Singapur)	sts.ap-southeast-1.amazonaws.com	Ja	Ja
Asien-Pazifik (Sydney)	sts.ap-southeast-2.amazonaws.com	Ja	Ja

Name der Region	Endpunkt	Standardmäßig aktiviert	Manuelles Aktivieren/ Deaktivieren
Asien-Pazifik (Tokio)	sts.ap-northeast-1.amazonaws.com	Ja	Ja
Kanada (Zentral)	sts.ca-central-1.amazonaws.com	Ja	Ja
Kanada West (Calgary)	sts.ca-west-1.amazonaws.com	Ja	Ja
China (Peking)	sts.cn-north-1.amazonaws.com.cn	Ja[2]	Nein
China (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	Ja[2]	Ja
Europa (Frankfurt)	sts.eu-central-1.amazonaws.com	Ja	Ja

Name der Region	Endpunkt	Standardmäßig aktiviert	Manuelles Aktivieren/ Deaktivieren
Europa (Irland)	sts.eu-west-1.amazonaws.com	Ja	Ja
Europa (London)	sts.eu-west-2.amazonaws.com	Ja	Ja
Europa (Milan)	sts.eu-south-1.amazonaws.com	Nein[1]	Nein
Europa (Paris)	sts.eu-west-3.amazonaws.com	Ja	Ja
Europa (Spain)	sts.eu-south-2.amazonaws.com	Nein[1]	Nein
Europa (Stockholm)	sts.eu-north-1.amazonaws.com	Ja	Ja

Name der Region	Endpunkt	Standardmäßig aktiviert	Manuelles Aktivieren/ Deaktivieren
Europa (Zürich)	sts.eu-central-2.amazonaws.com	 Nein[1]	 Nein
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	 Nein[1]	 Nein
Naher Osten (Bahrain)	sts.me-south-1.amazonaws.com	 Nein[1]	 Nein
Naher Osten (VAE)	sts.me-central-1.amazonaws.com	 Nein[1]	 Nein
Südamerika (São Paulo)	sts.sa-east-1.amazonaws.com	 Ja	 Ja

[1]Sie müssen die Region aktivieren, um sie nutzen zu können. Dies aktiviert AWS STS automatisch. Sie können AWS STS in diesen Regionen nicht manuell aktivieren oder deaktivieren.

[2]Für die Nutzung AWS in China benötigen Sie ein Konto und spezielle Anmeldeinformationen für China. AWS

AWS CloudTrail und regionale Endpunkte

Anrufe an regionale und globale Endpunkte werden im `tlsDetails`-Feld in AWS CloudTrail angemeldet. Anrufe an regionale Endpunkte, z. B.`us-east-2.amazonaws.com`, werden in der CloudTrail entsprechenden Region angemeldet. Aufrufe des globalen Endpunkts, z. B. `sts.amazonaws.com`, werden als Aufrufe eines globalen Services protokolliert. Ereignisse für globale AWS STS Endpunkte werden in us-east-1 protokolliert.

> **Note**
>
> `tlsDetails` kann nur für Services eingesehen werden, die dieses Feld unterstützen. Einzelheiten finden Sie im CloudTrail Benutzerhandbuch unter Dienste, die TLS unterstützen AWS CloudTrail
> Weitere Informationen finden Sie unter Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail.

Verwenden von Bearertoken

Für einige AWS Dienste benötigen Sie die Erlaubnis, ein AWS STS Service-Bearer-Token abzurufen, bevor Sie programmgesteuert auf ihre Ressourcen zugreifen können. Diese Services unterstützen ein Protokoll, bei dem Sie ein Bearertoken verwenden müssen, anstatt eine herkömmliche Signature Version 4-signierte Anforderung zu verwenden. Wenn Sie AWS API-Operationen ausführen AWS CLI , für die Inhaber-Token erforderlich sind, fordert der AWS Service in Ihrem Namen ein Inhaber-Token an. Der Service stellt Ihnen das Token zur Verfügung, mit dem Sie anschließend nachfolgende Vorgänge in diesem Service ausführen können.

AWS STS Service-Bearer-Token enthalten Informationen aus Ihrer ursprünglichen Prinzipalauthentifizierung, die sich auf Ihre Berechtigungen auswirken könnten. Diese Informationen können Auftraggeber-Tags, Sitzungs-Tags und Sitzungsrichtlinien enthalten. Die Zugriffsschlüssel-ID des Tokens beginnt mit dem Präfix `ABIA`. Auf diese Weise können Sie Vorgänge, die mithilfe von Service Bearer-Token ausgeführt wurden, in Ihren CloudTrail Protokollen identifizieren.

> **Important**
>
> Das Bearertoken kann nur für Aufrufe an den Service verwendet werden, der es generiert hat, und in der Region, in der es generiert wurde. Sie können das Bearertoken nicht verwenden, um Operationen in anderen Services oder Regionen auszuführen.

Ein Beispiel für einen Dienst, der Bearer-Token unterstützt, ist. AWS CodeArtifact Bevor Sie AWS CodeArtifact mit einem Paketmanager wie NPM, Maven oder PIP interagieren können, müssen Sie den Vorgang aufrufen. `aws codeartifact get-authorization-token` Dieser Vorgang gibt ein Bearer-Token zurück, mit dem Sie Operationen ausführen können. AWS CodeArtifact Alternativ können Sie den Befehl `aws codeartifact login` verwenden, der denselben Vorgang ausführt und den Client dann automatisch konfiguriert.

Wenn Sie eine Aktion in einem AWS Dienst ausführen, der ein Bearer-Token für Sie generiert, müssen Sie in Ihrer IAM-Richtlinie über die folgenden Berechtigungen verfügen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Ein Beispiel für Service Bearer-Token finden Sie unter Verwenden von identitätsbasierten Richtlinien für AWS CodeArtifact im AWS CodeArtifact-Benutzerhandbuch.

Beispielanwendungen, für die temporäre Anmeldeinformationen verwendet werden

Sie können AWS Security Token Service (AWS STS) verwenden, um vertrauenswürdige Benutzer zu erstellen und ihnen temporäre Sicherheitsanmeldeinformationen zur Verfügung zu stellen, mit denen der Zugriff auf Ihre AWS Ressourcen gesteuert werden kann. Mehr über AWS STS erfahren Sie unter Temporäre IAM Sicherheitsanmeldeinformationen. Um zu erfahren, wie Sie temporäre Sicherheitsanmeldeinformationen verwalten können, können Sie die folgenden Beispielanwendungen herunterladen, die vollständige Beispielszenarien implementieren: AWS STS

- Aktivierung des Verbunds für die AWS Verwendung von Windows Active Directory, ADFS und SAML 2.0. Zeigt, wie der Zugriff mithilfe eines Unternehmensverbunds unter Verwendung von Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0 und SAML (Security Assertion Markup Language) 2.0 an AWS delegiert wird.

- Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole: Zeigt, wie Sie einen benutzerdefinierten Verbund-Proxy erstellen, um Single Sign-On (SSO) zu aktivieren und es vorhandenen Active Directory-Benutzern zu ermöglichen, sich bei der AWS Management Console anzumelden.
- So verwenden Sie Shibboleth für Single Sign-On am. AWS Management Console . Zeigt, wie Sie mithilfe von Shibboleth und SAML Benutzern SSO-Zugriff (Single Sign-On) auf die AWS Management Console gewähren.

Beispiele für den OIDC-Verbund

Die folgenden Beispielanwendungen veranschaulichen die Verwendung von OIDCFederation mit Anbietern wie Login with Amazon, Amazon Cognito, Facebook oder Google. Sie können die Authentifizierung dieser Anbieter gegen temporäre AWS Sicherheitsanmeldedaten für den Zugriff auf Dienste eintauschen. AWS

- Amazon Cognito-Tutorials — Wir empfehlen, Amazon Cognito mit den AWS SDKs für die mobile Entwicklung zu verwenden. Amazon Cognito ist die einfachste Methode zur Identitätsverwaltung für mobile Apps. Außerdem werden hier zusätzliche Funktionen wie die Synchronisierung und geräteübergreifende Identitäten unterstützt. Weitere Informationen zu Amazon Cognito finden Sie unter Authentifizierung mit Amplify in der Amplify-Dokumentation.

Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole

Sie können Code schreiben und ausführen, um eine URL zu erstellen, über die Benutzer, die sich beim Netzwerk Ihrer Organisation anmelden, sicheren Zugriff auf die AWS Management Console haben. Die URL enthält ein Anmeldetoken, das Sie erhalten AWS und das den Benutzer authentifiziert. AWS Die daraus resultierende Konsolensitzung kann aufgrund eines Verbunds eine eindeutige `AccessKeyId` enthalten. Informationen zur Verwendung des Zugriffsschlüssels für die Verbundanmeldung anhand verwandter CloudTrail Ereignisse finden Sie unter Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail und AWS Management Console unter Anmeldeereignisse.

> **Note**
>
> Wenn Ihre Organisation einen Identitätsanbieter (IdP) verwendet, der mit SAML kompatibel ist, können Sie den Zugriff auf die Konsole einrichten, ohne dass Code geschrieben werden

muss. Dies funktioniert mit Anbietern wie Microsoft Active Directory Federation Services oder Open-Source-Shibboleth. Details hierzu finden Sie unter Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console.

Um den Benutzern Ihrer Organisation den Zugriff auf zu ermöglichen AWS Management Console, können Sie einen benutzerdefinierten Identity Broker erstellen, der die folgenden Schritte ausführt:

1. Überprüfen Sie, ob der Benutzer von Ihrem lokalen Identitätssystem authentifiziert ist.
2. Rufen Sie die Operationen AWS Security Token Service (AWS STS) AssumeRole(empfohlen) oder GetFederationTokenAPI auf, um temporäre Sicherheitsanmeldeinformationen für den Benutzer abzurufen. Weitere Informationen zu den unterschiedlichen Methoden, die Sie zum Übernehmen einer Rolle verwenden können, finden Sie unter Verwenden von IAM-Rollen. Weitere Informationen zur Übergabe optionaler Sitzungs-Tags bei Erhalt Ihrer Sicherheitsanmeldeinformationen finden Sie unter Sitzungs-Tags übergeben AWS STS.
   - Wenn Sie eine der `AssumeRole*` API-Operationen verwenden, um die temporären Sicherheitsanmeldeinformationen für eine Rolle zu erhalten, können Sie den Parameter `DurationSeconds` in Ihren Aufruf einschließen. Dieser Parameter gibt die Dauer Ihrer Rollensitzung von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle an. Wenn Sie `DurationSeconds` in einer `AssumeRole*`-Operation verwenden, müssen Sie das als IAM-Benutzer mit langfristigen Anmeldeinformationen aufrufen. Andernfalls schlägt der Aufruf des Verbund-Endpunkts in Schritt 3 fällt. Weitere Informationen dazu, wie Sie den maximalen Wert für eine Rolle anzeigen oder ändern können, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.
   - Wenn Sie die `GetFederationToken`-API-Operation verwenden, um die Anmeldeinformationen zu erhalten, können Sie den `DurationSeconds`-Parameter in Ihre Aufruf aufnehmen. Dieser Parameter gibt die Dauer Ihrer Rollensitzung an. Der Wert kann im Bereich zwischen 900 Sekunden (15 Minuten) und 129 600 Sekunden (36 Stunden) liegen. Sie können diesen API-Aufruf nur mit den langfristigen AWS Sicherheitsanmeldedaten eines IAM-Benutzers ausführen. Sie können diese Aufrufe auch mit Root-Benutzer des AWS-Kontos Anmeldeinformationen tätigen, wir empfehlen dies jedoch nicht. Wenn Sie diesen Aufruf als Stammbenutzer tätigen, dauert die Standardsitzung eine Stunde. Sie können auch eine Sitzung zwischen 900 Sekunden (15 Minuten) und 3 600 Sekunden (eine Stunde) festlegen.
3. Rufen Sie den AWS Verbundendpunkt auf und geben Sie die temporären Sicherheitsanmeldedaten ein, um ein Anmeldetoken anzufordern.
4. Erstellen Sie eine URL für die Konsole, die das Token enthält:

- Wenn Sie eine der `AssumeRole*`-API-Operationen in Ihrer URL verwenden, können Sie den `SessionDuration`-HTTP-Parameter einschließen. Dieser Parameter gibt die Dauer der Konsolensitzung an, von 900 Sekunden (15 Minuten) bis 43 200 Sekunden (12 Stunden).
- Wenn Sie die `GetFederationToken`-API-Operation in Ihrer URL verwenden, können Sie den `DurationSeconds`-Parameter einschließen. Dieser Parameter gibt die Dauer der verbundenen Konsolensitzung an. Der Wert kann im Bereich zwischen 900 Sekunden (15 Minuten) und 129 600 Sekunden (36 Stunden) liegen.

> **Note**
>
> - Verwenden Sie nicht den `SessionDuration`-HTTP-Parameter, wenn Sie die temporären Anmeldeinformationen mit `GetFederationToken` abrufen. Dadurch schlägt die Operation fehl.
> - Die Verwendung der Anmeldeinformationen für eine Rolle zur Übernahme einer anderen Rolle wird als Verketten von Rollen bezeichnet. Wenn Sie die Verkettung von Rollen verwenden, sind Ihre neuen Anmeldeinformationen auf eine maximale Dauer von einer Stunde begrenzt. Wenn Sie Rollen verwenden, um Berechtigungen für Anwendungen zu erteilen, die auf EC2-Instances ausgeführt werden, unterliegen diese Anwendungen nicht dieser Einschränkung.

5. Geben Sie die URL an den Benutzer weiter oder rufen Sie sie im Namen des Benutzers auf.

Die vom Verbund-Endpunkt bereitgestellte URL ist 15 Minuten lang gültig, nachdem sie erstellt wurde. Dies unterscheidet sich von der Dauer (in Sekunden) der Sitzung mit temporären Sicherheitsanmeldeinformationen, die mit der URL verbunden ist. Diese Anmeldeinformationen sind für die Dauer gültig, die Sie beim Erstellen angegeben haben, beginnend ab dem Zeitpunkt der Erstellung.

> **Important**
>
> Die URL gewährt Zugriff auf Ihre AWS Ressourcen über die, AWS Management Console sofern Sie in den zugehörigen temporären Sicherheitsanmeldedaten die entsprechenden Berechtigungen aktiviert haben. Aus diesem Grund sollten Sie die URL geheim halten. Wir empfehlen, die URL über eine sichere Umleitung zurückzugeben, z. B. mittels eines HTTP-Antwortstatuscodes 302 über eine SSL-Verbindung. Weitere Informationen über den HTTP-Antwortstatuscode 302 finden Sie unter RFC 2616, section 10.3.3.

Um diese Aufgaben zu erledigen, können Sie die HTTPS Query API für AWS Identity and Access Management (IAM) und die AWS Security Token Service (AWS STS) verwenden. Sie können aber auch Programmiersprachen, wie Java, Ruby oder C#, zusammen mit dem entsprechenden AWS -SDK, verwenden. Jede dieser Methoden wird in den folgenden Themen beschrieben.

Themen

- Beispielcode unter Verwendung von IAM-Abfrage-API-Operationen
- Beispielcode unter Verwendung von Python
- Beispielcode unter Verwendung von Java
- Beispiel für das Erstellen einer URL (Ruby)

Beispielcode unter Verwendung von IAM-Abfrage-API-Operationen

Sie können eine URL erstellen, die Ihren Verbundbenutzern direkten Zugriff auf die AWS Management Console gewährt. Diese Aufgabe verwendet die IAM- und AWS STS HTTPS-Abfrage-API. Weitere Informationen zum Erstellen von Abfrageanforderungen finden Sie unter Erstellen von Abfrageanforderungen.

> **Note**
>
> Das folgende Verfahren enthält Beispiele für Textzeichenfolgen. Zur Erhöhung der Lesbarkeit wurden Zeilenumbrüche zu einigen der längeren Beispiele hinzugefügt. Wenn Sie diese Zeichenfolgen zur eigenen Verwendung erstellen, sollten Sie keine Zeilenumbrüche setzen.

Um einem Verbundbenutzer Zugriff auf Ihre Ressourcen zu gewähren AWS Management Console

1. Authentifizieren Sie den Benutzer in Ihrem Identitäts- und Autorisierungssystem.
2. Rufen Sie temporäre Sicherheitsanmeldeinformationen für den Benutzer ab. Die temporären Anmeldeinformationen bestehen aus einer Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sitzungs-Token. Weitere Informationen über das Erstellen temporärer Anmeldeinformationen finden Sie unter Temporäre IAM Sicherheitsanmeldeinformationen.

   Um temporäre Anmeldeinformationen zu erhalten, rufen Sie entweder die AWS STS AssumeRoleAPI (empfohlen) oder die GetFederationTokenAPI auf. Weitere Informationen zu

den Unterschieden zwischen diesen API-Vorgängen finden Sie im AWS Sicherheitsblog unter Grundlegendes zu den API-Optionen für die sichere Delegierung des Zugriffs auf Ihr AWS Konto.

> **Important**
>
> Wenn Sie die GetFederationTokenAPI verwenden, um temporäre Sicherheitsanmeldedaten zu erstellen, müssen Sie die Berechtigungen angeben, die die Anmeldeinformationen dem Benutzer gewähren, der die Rolle übernimmt. Für alle API-Operationen, die mit `AssumeRole*` beginnen, verwenden Sie eine IAM-Rolle, um Berechtigungen zuzuweisen. Für die anderen API-Operationen hängt der Mechanismus von der API ab. Weitere Details finden Sie unter Steuern von Berechtigungen für temporäre Sicherheitsanmeldeinformationen. Wenn Sie die `AssumeRole*`-API-Operationen verwenden, müssen Sie sie außerdem als IAM-Benutzer mit langfristigen Anmeldeinformationen aufrufen. Andernfalls schlägt der Aufruf des Verbund-Endpunkts in Schritt 3 fällt.

3. Nachdem Sie die temporären Sicherheitsanmeldeinformationen abgerufen haben, integrieren Sie sie in eine JSON-Sitzungs-Zeichenfolge, um sie geben ein Anmelde-Token zu tauschen. Im folgenden Beispiel wird gezeigt, wie Sie die Anmeldeinformationen codieren. Ersetzen Sie den Platzhaltertext durch die entsprechenden Werte aus den Anmeldeinformationen, die Sie im vorherigen Schritt erhalten haben.

```
{"sessionId":"*** temporary access key ID ***",
"sessionKey":"*** temporary secret access key ***",
"sessionToken":"*** session token ***"}
```

4. Führen Sie eine URL-Codierung der Sitzungs-Zeichenfolge aus dem vorherigen Schritt durch. Da die Informationen, die Sie codieren, sensibel sind, raten wir davon ab, einen Webservice für diese Codierung zu verwenden. Verwenden Sie stattdessen ein lokal installiertes Feature in Ihrem Entwicklungs-Toolkits, um diese Informationen sicher zu codieren. Sie können die Funktion `urllib.quote_plus` in Python, die Funktion `URLEncoder.encode`in Java oder die Funktion `CGI.escape` in Ruby verwenden. Beispiele finden Sie an späterer Stelle in diesem Thema.

5. > **Note**
>
> AWS unterstützt hier POST-Anfragen.

Senden Sie Ihre Anfrage an den AWS Föderationsendpunkt:

`https://region-code.signin.aws.amazon.com/federation`

Eine Liste möglicher Werte für *region-code* finden Sie in der Spalte Region unter AWS Sign-In endpoints (-Anmelde-Endpunkte). Sie können optional den standardmäßigen Verbindungsendpunkt für AWS die Anmeldung verwenden:

`https://signin.aws.amazon.com/federation`

Die Anforderung muss die `Action`- und `Session`-Parameter enthalten und (optional) – wenn Sie eine `AssumeRole*`-API-Operation verwendet haben – einen `SessionDuration`-HTTP-Parameter, wie im folgenden Beispiel gezeigt.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> Die folgenden Anweisungen in diesem Schritt funktionieren nur mithilfe von GET-Anforderungen.

Der `SessionDuration`-HTTP-Parameter gibt die Dauer der Konsolensitzung an. Dies ist von der Dauer der temporären Anmeldeinformationen, die Sie mit dem `DurationSeconds`-Parameter angeben, unabhängig. Sie können für `SessionDuration` einen maximalen Wert von 43200 (12 Stunden) festlegen. Wenn der `SessionDuration` Parameter fehlt, wird für die Sitzung standardmäßig die Dauer der Anmeldeinformationen verwendet, die Sie AWS STS in Schritt 2 abgerufen haben (standardmäßig eine Stunde). Details zur Festlegung der Dauer unter Verwendung des Parameters `DurationSeconds` finden Sie in der -Dokumentation für die `AssumeRole`-API. Die Möglichkeit zum Erstellen einer Konsolensitzung, die länger als eine Stunde dauert, ist eine intrinsische `getSigninToken`-Operation des Verbund-Endpunkts.

> **Note**
>
> - Verwenden Sie nicht den `SessionDuration`-HTTP-Parameter, wenn Sie die temporären Anmeldeinformationen mit `GetFederationToken` abrufen. Dadurch schlägt die Operation fehl.
> - Die Verwendung der Anmeldeinformationen für eine Rolle zur Übernahme einer anderen Rolle wird als Verketten von Rollen bezeichnet. Wenn Sie die Verkettung von Rollen verwenden, sind Ihre neuen Anmeldeinformationen auf eine maximale Dauer von einer Stunde begrenzt. Wenn Sie Rollen verwenden, um Berechtigungen für Anwendungen zu erteilen, die auf EC2-Instances ausgeführt werden, unterliegen diese Anwendungen nicht dieser Einschränkung.

Wenn Sie Konsolensitzungen mit einer erweiterten Dauer aktivieren, erhöhen Sie das Risiko einer Aufdeckung der Anmeldeinformationen. Um dieses Risiko zu minimieren, können Sie die aktiven Konsolensitzungen für jede Rolle direkt deaktivieren, indem Sie Revoke Sessions (Sitzungen aufheben) auf der Seite Role Summary (Rollenübersicht) in der IAM-Konsole wählen. Weitere Informationen finden Sie unter Widerrufen der temporären Sicherheitsanmeldeinformationen für IAM-Rollen.

Das folgende Beispiel zeigt, wie die Anforderung aussehen kann. Die Zeilen sind hier für bessere Lesbarkeit gebrochen, die Zeichenfolge sollte aber in einer Zeile übermittelt werden.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

Die Antwort des Verbund-Endpunkts ist ein JSON-Dokument mit einem `SigninToken`-Wert. Er sollte wie im folgenden Beispiel aussehen.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS unterstützt hier POST-Anfragen.

Schließlich erstellen Sie die URL, die Ihre Verbundbenutzer für den Zugriff auf die verwenden können AWS Management Console. Die URL ist derselbe Verbund-URL-Endpunkt, den Sie in Step 5 verwendet haben, plus die folgenden Parameter:

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> Die folgenden Anweisungen in diesem Schritt funktionieren nur mithilfe von GET-API.

Das folgende Beispiel zeigt, wie die endgültige URL aussehen könnte. Die URL ist ab dem Zeitpunkt, zu dem sie erstellt wurde, 15 Minuten lang gültig. Die temporären Sicherheitsanmeldeinformationen und die in der URL eingebettete Konsolensitzung ist für die Dauer gültig, die Sie bei der ersten Anforderung im `SessionDuration`-HTTP-Parameter angegeben haben.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
```

```
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Beispielcode unter Verwendung von Python

Das folgende Beispiel zeigt, wie Sie mit Python programmgesteuert eine URL erstellen können, die Verbundbenutzern direkten Zugriff auf die AWS Management Console gewährt. Es gibt zwei Beispiele:

- Föderieren Sie über GET-Anfragen an AWS
- Verbinden Sie sich über POST-Anfragen an AWS

Beide Beispiele verwenden die AssumeRoleAPI AWS SDK for Python (Boto3)und, um temporäre Sicherheitsanmeldedaten abzurufen.

GET-Anforderungen verwenden

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your AWS-Konto,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
```

```
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
```

```
print (request_url)
```

GET-Anforderungen verwenden

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your A AWS-Konto,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
```

```
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
```

```
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)

driver = webdriver.Chrome()
driver.execute_script(jsrequest);
```

Beispielcode unter Verwendung von Java

Das folgende Beispiel zeigt, wie Sie mit Java programmgesteuert eine URL erstellen können, die Verbundbenutzern direkten Zugriff auf die gewährt AWS Management Console. Das folgende Codefragment verwendet das AWS SDK for Java.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
```

```
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Action\":\"sns:*\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
                           URLEncoder.encode(sessionJson,"UTF-8");
```

```
URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Beispiel für das Erstellen einer URL (Ruby)

Das folgende Beispiel zeigt, wie Sie mit Ruby programmgesteuert eine URL erstellen können, die Verbundbenutzern direkten Zugriff auf die gewährt AWS Management Console. Dieses Codefragment verwendet das AWS SDK for Ruby.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
```

```
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",\"Statement\":{\"Effect\":\"Allow\",\"Action\":
\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read
```

```
# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Zusätzliche Ressourcen für temporäre Sicherheitsanmeldeinformationen

Folgende Szenarien und Anwendungen führen Sie durch die Verwendungsmöglichkeiten von temporären Sicherheitsanmeldeinformationen:

- So integrieren Sie sich AWS STS SourceIdentity mit Ihrem Identitätsanbieter. In diesem Beitrag erfahren Sie, wie Sie das AWS STS `SourceIdentity` Attribut einrichten, wenn Sie Okta, Ping oder OneLogin als Ihren IdP verwenden.
- OIDC-Föderation. In diesem Abschnitt wird beschrieben, wie Sie IAM-Rollen konfigurieren, wenn Sie den OIDC-Verbund und die API verwenden. `AssumeRoleWithWebIdentity`
- Konfigurieren eines MFA-geschützten API-Zugriffs. In diesem Thema wird erläutert, wie Sie mithilfe von Rollen die Multi-Factor Authentication (MFA) verlangen können, um die sensiblen API-Aktionen in Ihrem Konto zu schützen.

Weitere Informationen zu Richtlinien und Berechtigungen AWS finden Sie in den folgenden Themen:

- Zugriffsmanagement für AWS Ressourcen
- Auswertungslogik für Richtlinien.
- Verwalten von Zugriffsberechtigungen für Ihre Amazon-S3-Ressourcen im Benutzerhandbuch für Amazon Simple Storage Service.
- Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

Markieren von IAM-Ressourcen

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie einer AWS -Ressource zuweisen können. Jedes Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel (z. B. `CostCenter`, `Environment`, `Project` oder `Purpose`).
- einem optionalen Feld, das als Tag-Wert bezeichnet wird (z. B. `111122223333`, `Production` oder ein Team-Name). Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge.

Zusammen werden sie als Schlüssel-Wert-Paare bezeichnet. Informationen zu Beschränkungen für die Anzahl der Tags, die Sie für IAM-Ressourcen haben können, finden Sie unter IAM und Kontingente AWS STS.

> **Note**
>
> Einzelheiten zur Berücksichtigung der Groß- und Kleinschreibung bei Tag-Schlüsseln und Tag-Schlüsselwerten finden Sie unter Case sensitivity.

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen miteinander verknüpft sind. Sie können beispielsweise das gleiche Tag einer IAM-Rolle zuweisen, das Sie einem Amazon S3-Bucket zuweisen. Weitere Informationen zu Tagging-Strategien finden Sie im Benutzerhandbuch zum Markieren von AWS Ressourcen.

Sie können Ihre IAM-Ressourcen nicht nur mit Tags identifizieren, organisieren und verfolgen, sondern auch Tags in IAM-Richtlinien verwenden, um zu kontrollieren, wer Ihre Ressourcen anzeigen und mit ihnen interagieren kann. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer zusammenschließen. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Wählen Sie eine Konvention zur Benennung von AWS Tags

Wenn Sie mit dem Anfügen von Tags an Ihre IAM-Ressourcen beginnen möchten, wählen Sie Ihre Tag-Benennungskonvention sorgfältig. Wenden Sie dieselbe Konvention auf alle Ihre AWS Tags an. Dies ist besonders wichtig, wenn Sie Tags in Richtlinien verwenden, um den Zugriff auf AWS Ressourcen zu kontrollieren. Wenn Sie bereits Tags in AWS verwenden, überprüfen Sie Ihre Namenskonvention und passen Sie sie entsprechend an.

> **Note**
>
> Wenn Ihr Konto Mitglied von ist AWS Organizations, finden Sie unter Tag-Richtlinien im Benutzerhandbuch für Organizations weitere Informationen zur Verwendung von Stichwörtern in Organizations.

Best Practices für Tag-Benennung

Dies sind einige Best Practices und Namenskonventionen für Tags.

Stellen Sie sicher, dass Tag-Namen konsistent verwendet werden. Zum Beispiel sind die Tags `CostCenter` und `costcenter` unterschiedlich, sodass eines möglicherweise als Kostenverteilungs-Tag für Finanzanalysen und Berichte konfiguriert ist und das andere möglicherweise nicht. In ähnlicher Weise wird das `Name` Tag für viele Ressourcen in der AWS Konsole angezeigt, das `name` Tag jedoch nicht. Einzelheiten zur Berücksichtigung der Groß- und Kleinschreibung bei Tag-Schlüsseln und Tag-Schlüsselwerten finden Sie unter Case sensitivity.

Eine Reihe von Tags ist von verschiedenen AWS Diensten vordefiniert AWS oder wird automatisch von diesen erstellt. Bei vielen AWS-definierten Tagnamen werden ausschließlich Kleinbuchstaben verwendet, wobei Wörter im Namen durch Bindestriche voneinander getrennt werden, und Präfixe, um den Quelldienst für das Tag zu identifizieren. Beispielsweise:

- `aws:ec2spot:fleet-request-id` identifiziert den Amazon EC2 Spot Instance Request, der die Instance gestartet hat.
- `aws:cloudformation:stack-name`identifiziert den AWS CloudFormation Stapel, der die Ressource erstellt hat.
- `elasticbeanstalk:environment-name` identifiziert die Anwendung, die die Ressource erstellt hat.

Benennen Sie Ihre Tags in Kleinbuchstaben, mit Bindestrichen, die Wörter trennen, und einem Präfix, das den Namen der Organisation oder den abgekürzten Namen identifiziert. Für ein fiktives Unternehmen mit dem Namen könnten Sie AnyCompanybeispielsweise Tags definieren wie:

- `anycompany:cost-center`, um den internen Cost Center-Code zu identifizieren
- `anycompany:environment-type`, um festzustellen, ob es sich bei der Umgebung um Entwicklung, Test oder Produktion handelt
- `anycompany:application-id`, um die Anwendung zu identifizieren, für die die Ressource erstellt wurde

Das Präfix stellt sicher, dass Tags eindeutig als von Ihrer Organisation definiert wurden und nicht von AWS einem Drittanbieter-Tool, das Sie möglicherweise verwenden. Die Verwendung von Kleinbuchstaben mit Bindestrichen für Trennzeichen vermeidet Verwirrung bei der Großschreibung eines Tag-Namens. Zum Beispiel ist es einfacher, sich `anycompany:project-id` zu merken als `ANYCOMPANY:ProjectID`, `anycompany:projectID` oder`Anycompany:ProjectId`.

Regeln für das Tagging in IAM und AWS STS

Eine Reihe von Konventionen reguliert die Erstellung und Anwendung von Tags in IAM und AWS STS.

Benennen von Tags

Beachten Sie bei der Formulierung einer Konvention zur Benennung von Tags für IAM-Ressourcen, Sitzungen mit übernommener AWS STS Rolle und Verbundbenutzersitzungen die folgenden Konventionen: AWS STS

Zeichenanforderungen – Tag-Schlüssel und -Werte können eine beliebige Kombination aus Buchstaben, Zahlen, Leerzeichen und den Symbolen _ . : / = + - @ enthalten.

Groß-/Kleinschreibung – Groß-/Kleinschreibung für Tag-Schlüssel hängt vom Typ der getaggten IAM-Ressource ab. Tag-Schlüsselwerte für IAM-Benutzer und IAM-Rollen unterscheiden nicht zwischen Groß-/Kleinschreibung, die Groß-/Kleinschreibung wird jedoch berücksichtigt. Dies bedeutet, dass Sie keine separaten **`Department`**- und **`department`**-Tag-Schlüssel haben können. Wenn Sie einen Benutzer mit dem **`Department=finance`**-Tag markiert haben und Sie das **`department=hr`**-Tag hinzufügen, wird das erste Tag ersetzt. Ein zweites Tag wird nicht hinzugefügt.

Bei anderen IAM-Ressourcentypen wird bei Tag-Schlüsselwerten die Groß-/Kleinschreibung unterscheiden. Das bedeutet, dass Sie separate **`Costcenter`**- und **`costcenter`**-Tag-Schlüssel

haben können. Wenn Sie beispielsweise eine von Kunden verwaltete Richtlinie mit dem **`Costcenter = 1234`**-Tag markiert haben und das **`costcenter = 5678`**-Tag hinzufügen, verfügt die Richtlinie über die **`Costcenter`**- und die **`costcenter`**-Tag-Schlüssel.

Als Best Practice empfehlen wir Ihnen, die Verwendung ähnlicher Tags bei inkonsistenter Groß-/Kleinschreibung zu vermeiden. Wir empfehlen Ihnen, sich für eine einheitliche Schreibweise der Tag-Benennungen zu entscheiden und diese Strategie für alle Ressourcentypen umzusetzen. Weitere Informationen zu bewährten Methoden für das Tagging finden Sie unter Tagging Resources im. AWS Allgemeine AWS-Referenz

Die folgenden Listen zeigen die Unterschiede in der Groß-/Kleinschreibung für Tag-Schlüssel, die an IAM-Ressourcen angehängt sind.

Tag-Schlüsselwerte unterscheiden nicht zwischen Groß-/Kleinschreibung:

- IAM-Rollen
- IAM-Benutzer

Bei Tag-Schlüsselwerten wird die Groß-/Kleinschreibung berücksichtigt.

- Kundenverwaltete Richtlinien
- Instance-Profile
- OpenID-Connect-Identitätsanbieter
- SAML-Identitätsanbieter
- Serverzertifikate
- Virtuelle MFA-Geräte

Darüber hinaus gelten die folgenden Regeln:

- Sie können keinen Tag-Schlüssel oder -Wert erstellen, der mit dem Text **`aws:`** beginnt. Dieses Tag-Präfix ist für den AWS internen Gebrauch reserviert.
- Sie können ein Tag mit einem leeren Wert erstellen, wie z. B. **`phoneNumber =`** . Sie können keinen leeren Tag-Schlüssel erstellen.
- Sie können nicht mehrere Werte in einem einzigen Tag angeben. Sie können jedoch eine benutzerdefinierte mehrwertige Struktur in dem einzelnen Wert erstellen. Nehmen Sie beispielsweise an, dass der Benutzer Zhang im Entwicklungsteam und dem QA-Team arbeitet.

Wenn Sie das **`team = Engineering`**-Tag und dann das **`team = QA`** Tag anfügen, ändern Sie den Wert des Tags von **`Engineering`** zu **`QA`**. Stattdessen können Sie mehrere Werte in einem einzigen Tag mit einem benutzerdefinierten Trennzeichen einfügen. In diesem Beispiel könnten Sie das **`team = Engineering:QA`**-Tag Zhang anfügen.

> **Note**
>
> Um den Zugriff auf die Entwickler in diesem Beispiel mit dem **`team`**-Tag zu steuern, müssen Sie eine Richtlinie erstellen, die jede Konfiguration zulässt, die **`Engineering`** einschließlich **`Engineering:QA`** enthalten könnte. Weitere Informationen zur Verwendung von Tags in Richtlinien finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Anwenden und Bearbeiten von Tags

Beachten Sie die folgenden Konventionen beim Anfügen von Tags zu IAM-Ressourcen:

- Sie können die meisten IAM-Ressourcen markieren, jedoch keine Gruppen, angenommene Rollen, Zugriffsberichte oder hardwarebasierte MFA-Geräte.
- Sie können den Tag-Editor nicht verwenden, um IAM-Ressourcen zu markieren. Der Tag-Editor unterstützt keine IAM-Tags. Informationen zur Verwendung des Tag-Editors mit anderen Services finden Sie unter Arbeiten mit dem Tag-Editor im AWS Resource Groups -Leitfaden.
- Um eine IAM-Ressource zu markieren, müssen Sie über spezifische Berechtigungen verfügen. Um Ressourcen zu markieren oder die Markierung zu entfernen, müssen Sie auch über die Berechtigung verfügen, Tags aufzulisten. Weitere Informationen finden Sie in der Liste der Themen für jede IAM-Ressource am Ende dieser Seite.
- Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.
- Sie können dasselbe Tag auf mehrere IAM-Ressourcen anwenden. Angenommen, Sie haben eine Abteilung namens Namen `AWS_Development` mit 12 Mitarbeitern. Sie können 12 Benutzer und eine Rolle mit dem Tag-Schlüssel **`department`** und dem Wert **`awsDevelopment`** (**`department = awsDevelopment`**) haben. Darüber hinaus können Sie das gleiche Tag für Ressourcen in anderen Services verwenden, die Markierungen unterstützen.
- IAM-Entitäten (Benutzer oder Rollen) können nicht mehrere Instances desselben Tag-Schlüssels haben. Wenn Sie z. B. einen Benutzer mit dem Tag-Schlüssel-Wert-Paar **`costCenter = 1234`**

haben, können Sie das Tag-Schlüssel-Wert-Paar **`costCenter = 5678`** anfügen. IAM aktualisiert den Wert des **`costCenter`**-Tag auf **`5678`**.

- Um ein Tag zu bearbeiten, das einer IAM-Entität (Benutzer oder Rolle) angefügt ist, fügen Sie ein Tag mit einem neuen Wert an, um das vorhandene Tag zu überschreiben. Angenommen, Sie haben einen Benutzer mit dem Tag-Schlüsselwertpaar **`department = Engineering`**. Wenn Sie den Benutzer in die QA-Abteilung verschieben möchten, dann können Sie das **`department = QA`**-Tag-Schlüsselwertpaar an den Benutzer anfügen. Dadurch wird der **`Engineering`**-Wert des **`department`**-Tag-Schlüssels mit dem **`QA`**-Wert ersetzt.

Themen

Markieren von IAM-Benutzern

Sie können IAM-Tag-Schlüsselwertpaare verwenden, um einem IAM-Benutzer benutzerdefinierte Attribute hinzuzufügen. Wenn Sie beispielsweise einem Benutzer Standortinformationen hinzufügen, können Sie den Tag-Schlüssel **`location`** und den Tag-Wert **`us_wa_seattle`** hinzufügen. Alternativ könnten Sie drei getrennte Standort-Tag-Schlüsselwertpaare verwenden: **`loc-country = us`**, **`loc-state = wa`** und **`loc-city = seattle`**. Sie können Tags verwenden, um den Zugriff eines Benutzers auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einem Benutzer verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer verbinden. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Erforderliche Berechtigungen für das Markieren von IAM-Benutzern

Sie müssen Berechtigungen konfigurieren, damit ein IAM-Benutzer andere Benutzer markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

So erlauben Sie einem IAM-Benutzer das Hinzufügen, Auflisten oder Entfernen eines Tags für einen bestimmten Benutzer

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für den IAM-Benutzer hinzu, der Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *<username>* durch den Namen des Benutzers, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

So erlauben Sie einem IAM-Benutzer Tags selbst zu verwalten

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für Benutzer hinzu, um Benutzern zu erlauben, ihre eigenen Tags zu verwalten. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
```

```
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

So erlauben Sie einem IAM-Benutzer, einem bestimmten Benutzer ein Tag hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für den IAM-Benutzer hinzu, der Tags für einen bestimmten Benutzer hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagUser`-Aktion erfordert, dass Sie auch die `iam:ListUserTags`-Aktion einbeziehen.

Zur Verwendung dieser Richtlinie ersetzen Sie *<username>* durch den Namen des Benutzers, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, FullAccess um vollen Zugriff auf IAM zu gewähren.

Verwalten von Tags auf IAM-Benutzern (Konsole)

Sie können Tags für IAM-Benutzer über die AWS Management Console verwalten.

Verwalten von Tags auf Benutzern (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. Wählen Sie im Navigationsbereich der Konsole Users (Benutzer) aus und wählen Sie dann den Namen des Benutzers aus, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Tags aus und schließen Sie eine der folgenden Aktionen ab:
   - Wählen Sie Add new tag (Neues Tag hinzufügen), wenn der Benutzer noch nicht über Tags verfügt.
   - Wählen Sie Manage tags (Tags verwalten), um den vorhandenen Satz von Tags zu verwalten.
4. Fügen Sie Tags hinzu oder entfernen Sie sie, um den Satz von Tags abzuschließen. Wählen Sie dann Save changes (Änderungen speichern).

Verwaltung von Tags für IAM-Benutzer (AWS CLI oder AWS API)

Sie können Tags für IAM-Benutzer auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für IAM-Benutzer zu verwalten.

Um die Tags aufzulisten, die derzeit einem IAM-Benutzer (AWS CLI oder AWS einer API) zugeordnet sind

- AWS CLI: Was ich bin list-user-tags
- AWS API: ListUserTags

Um Tags an einen IAM-Benutzer (AWS CLI oder eine AWS API) anzuhängen

- AWS CLI: aws iam tag-user
- AWS API: TagUser

Um Tags von einem IAM-Benutzer (AWS CLI oder einer AWS API) zu entfernen

- AWS CLI: aws iam untag-user
- AWS API: UntagUser

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Markieren von IAM-Rollen

Sie können -Tag-Schlüsselwertpaare verwenden, um einer IAM-Rolle benutzerdefinierte Attribute hinzuzufügen. Wenn Sie beispielsweise einer Rolle Standortinformationen hinzufügen, können Sie den Tag-Schlüssel **`location`** und den Tag-Wert **`us_wa_seattle`** hinzufügen. Alternativ könnten Sie drei getrennte Standort-Tag-Schlüsselwertpaare verwenden: **`loc-country = us`**, **`loc-state = wa`** und **`loc-city = seattle`**. Sie können Tags verwenden, um den Zugriff einer Rolle auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einer Rolle verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer verbinden. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Erforderliche Berechtigungen für das Markieren von IAM-Rollen

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Rolle andere Entitäten (Benutzer oder Rolle) markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListRoleTags`
- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

So erlauben Sie einer IAM-Rolle das Hinzufügen, Auflisten oder Entfernen eines Tags für einen bestimmten Benutzer

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Rolle hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *`<username>`* durch den Namen des Benutzers, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

So erlauben Sie einer IAM-Rolle das Hinzufügen eines Tags zu einem bestimmten Benutzer

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Rolle hinzu, die Tags für einen bestimmten Benutzer hinzufügen, jedoch nicht entfernen soll.

Zur Verwendung dieser Richtlinie ersetzen Sie *<username>* durch den Namen des Benutzers, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

So erlauben Sie einer IAM-Rolle das Hinzufügen, Auflisten oder Entfernen eines Tags für eine bestimmte Rolle

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Rolle hinzu, die Tags verwalten soll. Ersetzen Sie *<rolename>* durch den Namen der Rolle, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
```

```
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, FullAccess um vollen Zugriff auf IAM zu gewähren.

Verwalten von Tags auf IAM-Rollen (Konsole)

Sie können Tags für IAM-Rollen über die AWS Management Console verwalten.

Verwalten von Tags auf Rollen (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. Wählen Sie im Navigationsbereich der Konsole Rollen aus und wählen Sie dann den Namen der Rolle aus, die Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Tags aus und schließen Sie eine der folgenden Aktionen ab:
   - Wählen Sie Add new tag (Neues Tag hinzufügen), wenn die Rolle noch nicht über Tags verfügt.
   - Wählen Sie Manage tags (Tags verwalten), um den vorhandenen Satz von Tags zu verwalten.
4. Fügen Sie Tags hinzu oder entfernen Sie sie, um den Satz von Tags abzuschließen. Dann wählen Sie Save changes (Änderungen speichern) aus.

Verwaltung von Tags in IAM-Rollen (AWS CLI oder AWS API)

Sie können Tags für IAM-Rollen auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für IAM-Rollen zu verwalten.

Um die Tags aufzulisten, die derzeit mit einer IAM-Rolle (AWS CLI oder AWS API) verknüpft sind

- AWS CLI: war ich list-role-tags
- AWS API: ListRoleTags

Um Tags an eine IAM-Rolle (AWS CLI oder AWS API) anzuhängen

- AWS CLI: aws iam tag-role
- AWS API: TagRole

Um Tags aus einer IAM-Rolle (AWS CLI oder AWS API) zu entfernen

- AWS CLI: aws iam untag-role
- AWS API: UntagRole

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Markieren von vom Kunden verwalteten Richtlinien

Sie können IAM-Tag-Schlüsselwertpaare verwenden, um Ihren vom Kunden verwalteten Richtlinien benutzerdefinierte Attribute hinzuzufügen. Wenn Sie beispielsweise eine Richtlinie mit Abteilungsinformationen markieren, können Sie den Tag-Schlüssel **`Department`** und den Tag-Wert **`eng`** hinzufügen. Oder Sie möchten Richtlinien markieren, um anzugeben, dass sie für eine bestimmte Umgebung bestimmt sind, z. B. **`Environment = lab`**. Sie können Tags verwenden, um den Zugriff auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einer Ressource verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer verbinden. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Erforderliche Berechtigungen zum Markieren von vom Kunden verwalteten Richtlinien

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rollen) vom Kunden verwaltete Richtlinien markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListPolicyTags`

- `iam:TagPolicy`
- `iam:UntagPolicy`

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag einer vom Kunden verwalteten Richtlinie hinzuzufügen, aufzulisten oder zu entfernen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *<policyname>* durch den Namen der Richtlinie, deren Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag einer speziellen vom Kunden verwalteten Richtlinie hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags für eine bestimmte Richtlinie hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagPolicy`-Aktion erfordert, dass Sie auch die `iam:ListPolicyTags`-Aktion einbeziehen.

Um diese Richtlinie zu verwenden, ersetzen Sie *<policyname>* durch den Namen der Richtlinie, deren Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, FullAccess um vollen Zugriff auf IAM zu gewähren.

Verwalten von Tags für vom Kunden verwaltete IAM-Richtlinien (Konsole)

Sie können Tags für von IAM-Kunden verwaltete Richtlinien über die AWS Management Console verwalten.

Verwalten von Tags für von kundenverwalteten Richtlinien (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. Wählen Sie im Navigationsbereich der Konsole Richtlinien aus und wählen Sie dann den Namen der kundenverwalteten Richtlinie aus, die Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Tags und dann Manage tags (Tags verwalten) aus.
4. Fügen Sie Tags hinzu oder entfernen Sie sie, um den Satz von Tags abzuschließen. Wählen Sie dann Save changes (Änderungen speichern).

Verwaltung von Tags in von Kunden verwalteten IAM-Richtlinien (AWS CLI oder AWS API)

Sie können Tags für von IAM-Kunden verwaltete Richtlinien auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für vom Kunden verwaltete IAM-Richtlinien zu verwalten.

Um die Tags aufzulisten, die derzeit mit einer vom Kunden verwalteten IAM-Richtlinie (AWS CLI oder AWS API) verknüpft sind

- AWS CLI: Was ich bin list-policy-tags

- AWS API: ListPolicyTags

Um Tags an eine vom Kunden verwaltete IAM-Richtlinie (AWS CLI oder AWS API) anzuhängen

- AWS CLI: aws iam tag-policy
- AWS API: TagPolicy

Um Tags aus einer vom Kunden verwalteten IAM-Richtlinie (AWS CLI oder AWS API) zu entfernen

- AWS CLI: aws iam untag-policy
- AWS API: UntagPolicy

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Markieren von IAM-Identitätsanbietern

Sie können IAM-Tag-Schlüssel-Wert-Paare verwenden, um benutzerdefinierte Attribute zu IAM-Identitätsanbietern hinzuzufügen (). IdPs

Sie können Tags auch verwenden, AWS STS um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer zusammenschließen. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Weitere Informationen zum Tagging IdPs in IAM finden Sie in den folgenden Themen:

Themen

- Markieren von OpenID-Connect(OIDC)-Identitätsanbietern
- Markieren von IAM-SAML-Identitätsanbietern

Markieren von OpenID-Connect(OIDC)-Identitätsanbietern

Sie können -Tag-Schlüsselwertpaare verwenden, um IAM-OpenID-Connect(OIDC)-Identitätsanbietern benutzerdefinierte Attribute hinzuzufügen. Um beispielsweise einen IAM-OIDC-Identitätsanbieter zu identifizieren, können Sie den Tag-Schlüssel **`google`** und den Tag-Wert

`oidc` hinzufügen. Sie können Tags verwenden, um den Zugriff auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einem Objekt verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Erforderliche Berechtigungen zum Markieren von IAM-OIDC-Identitätsanbietern

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rolle) andere IAM-OIDC-Identitätsanbieter markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag für einen IAM-OIDC-Identitätsanbieter hinzuzufügen, aufzulisten oder zu entfernen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *<OIDC ProviderName >* durch den Namen des OIDC-Anbieters, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), einem bestimmten IAM-OIDC-Identitätsanbieter ein Tag hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags für einen bestimmten Identitätsanbieter hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagOpenIDConnectProvider`-Aktion erfordert, dass Sie auch die `iam:ListOpenIDConnectProviderTags`-Aktion einbeziehen.

Um diese Richtlinie zu verwenden, ersetzen Sie *<OIDC ProviderName >* durch den Namen des OIDC-Anbieters, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, um vollen Zugriff auf IAM FullAccess zu gewähren.

Verwalten von Tags auf IAM-OIDC-Identitätsanbietern (Konsole)

Sie können Tags für IAM-OIDC-Identitätsanbieter über AWS Management Console verwalten.

Verwalten der Tags auf OIDC-Identitätsanbietern (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. Wählen Sie im Navigationsbereich der Konsole Identitätsanbieter und anschließend den Namen des Identitätsanbieters aus, den Sie bearbeiten möchten.
3. Wählen Sie im Abschnitt Tags Managing Tags (Tags verwalten) aus und schließen Sie eine der folgenden Aktionen ab:

   - Wählen Sie Add Tag (Tag hinzufügen), wenn der OIDC-Identitätsanbieter noch keine Tags hat oder um ein neues Tag hinzuzufügen.
   - Bearbeiten Sie vorhandene Tag-Schlüssel und -Werte.

- Klicken Sie zum Entfernen eines Tags auf Remove tag (Tag entfernen).

4. Wählen Sie dann Save changes (Änderungen speichern).

Verwaltung von Tags bei IAM OIDC-Identitätsanbietern (oder API)AWS CLIAWS

Sie können Tags für IAM-OIDC-Identitätsanbieter auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für IAM OIDC-Identitätsanbieter zu verwalten.

Um die Tags aufzulisten, die derzeit an einen IAM-OIDC-Identitätsanbieter (oder eine API) angehängt sind AWS CLIAWS

- AWS CLI: aws iam -provider-Tags list-open-id-connect
- AWS ListOpenAPI: ID ConnectProviderTags

Um Tags an einen IAM OIDC-Identitätsanbieter (AWS CLI oder eine API) anzuhängen AWS

- AWS CLI: aws iam -provider tag-open-id-connect
- AWS API: ID TagOpen ConnectProvider

Um Tags von einem IAM OIDC-Identitätsanbieter (AWS CLI oder einer API) zu entfernen AWS

- AWS CLI: aws iam -provider untag-open-id-connect
- AWS API: ID UntagOpen ConnectProvider

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Markieren von IAM-SAML-Identitätsanbietern

Sie können IAM-Tag-Schlüsselwertpaare verwenden, um SAML-Identitätsanbietern benutzerdefinierte Attribute hinzuzufügen. Um beispielsweise einen Anbieter zu identifizieren, können Sie den Tag-Schlüssel **`okta`** und den Tag-Wert **`saml`** hinzufügen. Sie können Tags verwenden, um den Zugriff auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einem Objekt verknüpft

werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Erforderliche Berechtigungen zum Markieren von SAML-Identitätsanbietern

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rollen) SAML 2.0-basierte Identitätsanbieter () taggen kann. IdPs Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag für einen SAML-Identitätsanbieter hinzuzufügen, aufzulisten oder zu entfernen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *<SAML ProviderName >* durch den Namen des SAML-Anbieters, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), einem bestimmten SAML-Identitätsanbieter ein Tag hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags für einen bestimmten SAML-Anbieter hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagSAMLProvider`-Aktion erfordert, dass Sie auch die `iam:ListSAMLProviderTags`-Aktion einbeziehen.

Um diese Richtlinie zu verwenden, ersetzen Sie *<SAML ProviderName >* durch den Namen des SAML-Anbieters, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, um vollen Zugriff auf IAM FullAccess zu gewähren.

Verwalten von Tags auf IAM-SAML-Identitätsanbietern (Konsole)

Sie können Tags für IAM–SAML-Identitätsanbieter über AWS Management Console verwalten.

Verwalten der Tags auf SAML-Identitätsanbietern (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. Wählen Sie im Navigationsbereich der Konsole Identitätsanbieter und anschließend den Namen des SAML-Identitätsanbieters aus, den Sie bearbeiten möchten.
3. Wählen Sie im Abschnitt Tags Managing Tags (Tags verwalten) aus und schließen Sie eine der folgenden Aktionen ab:

   - Wählen Sie Add Tag (Tag hinzufügen), wenn der SAML-Identitätsanbieter noch keine Tags hat oder um ein neues Tag hinzuzufügen.
   - Bearbeiten Sie vorhandene Tag-Schlüssel und -Werte.

- Klicken Sie zum Entfernen eines Tags auf Remove tag (Tag entfernen).

4. Fügen Sie Tags hinzu oder entfernen Sie sie, um den Satz von Tags abzuschließen. Wählen Sie dann Save changes (Änderungen speichern).

Verwaltung von Tags auf IAM-SAML-Identitätsanbietern (oder API)AWS CLIAWS

Sie können Tags für IAM-SAML-Identitätsanbieter auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für IAM-SAML-Identitätsanbieter zu verwalten.

Um die Tags aufzulisten, die derzeit an einen SAML-Identitätsanbieter (AWS CLI oder eine API) angehängt sind AWS

- AWS CLI: als Ziel list-saml-provider-tags
- AWS API: listSAML ProviderTags

Um Tags an einen SAML-Identitätsanbieter (AWS CLI oder eine API) anzuhängen AWS

- AWS CLI: als Ziel tag-saml-provider
- AWS API: TagSamlProvider

Um Tags von einem SAML-Identitätsanbieter (oder einer API) zu entfernen AWS CLIAWS

- AWS CLI: als Ziel untag-saml-provider
- AWS API: UntagSamlProvider

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Kennzeichnen von Instance-Profilen für Amazon EC2-Rollen

Wenn Sie eine Amazon EC2-Instance starten, geben Sie eine mit der Instance zu verknüpfende IAM-Rolle an. Ein Instance-Profil ist ein Container für eine IAM-Rolle, mit dem eine Amazon EC2-Instance bei ihrem Start Rolleninformationen erhält. Sie können Instanzprofile taggen, wenn Sie die AWS API AWS CLI oder verwenden.

Sie können IAM-Tag-Schlüsselwertpaare verwenden, um einem Instance-Profil benutzerdefinierte Attribute hinzuzufügen. Wenn Sie beispielsweise einem Instance-Profil Abteilungsinformationen hinzufügen, können Sie den Tag-Schlüssel **`access-team`** und den Tag-Wert **`eng`** hinzufügen. Auf diese Weise erhalten Auftraggeber mit übereinstimmenden Tags Zugriff auf Instanceprofile mit demselben Tag. Sie könnten mehrere Tag-Schlüsselwertpaare verwenden, um ein Team und ein Projekt anzugeben: **`access-team = eng`** und **`project = peg`**. Sie können Tags verwenden, um den Zugriff eines Benutzers auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einem Benutzer verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer zusammenschließen. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Erforderliche Berechtigungen für das Markieren von Instance-Profilen

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rolle) andere Instance-Profile markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag für ein Instance-Profil hinzuzufügen, aufzulisten oder zu entfernen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *< InstanceProfileName >* durch den Namen des Instanzprofils, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
```

```
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), einem bestimmten Instance-Profil ein Tag hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags für ein bestimmtes Instance-Profil hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagInstanceProfile`-Aktion erfordert, dass Sie auch die `iam:ListInstanceProfileTags`-Aktion einbeziehen.

Um diese Richtlinie zu verwenden, ersetzen Sie *< InstanceProfileName >* durch den Namen des Instanzprofils, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, FullAccess um vollen Zugriff auf IAM zu gewähren.

Verwaltung von Tags in Instanzprofilen (AWS CLI oder AWS API)

Sie können Tags für Instance-Profile auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für Instanzprofile zu verwalten.

Um die Tags aufzulisten, die derzeit an ein Instanzprofil (AWS CLI oder eine AWS API) angehängt sind

- AWS CLI: war ich list-instance-profile-tags
- AWS API: ListInstanceProfileTags

Um Tags an ein Instanzprofil (AWS CLI oder eine AWS API) anzuhängen

- AWS CLI: war ich tag-instance-profile
- AWS API: TagInstanceProfile

Um Tags aus einem Instanzprofil (AWS CLI oder einer AWS API) zu entfernen

- AWS CLI: war ich untag-instance-profile
- AWS API: UntagInstanceProfile

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Markieren von Serverzertifikaten

Wenn Sie IAM zur Verwaltung von SSL/TLS-Zertifikaten verwenden, können Sie Serverzertifikate in IAM mithilfe der API oder kennzeichnen. AWS CLI AWS Für Zertifikate in einer Region, die von AWS Certificate Manager (ACM) unterstützt wird, empfehlen wir, ACM anstelle von IAM für die Bereitstellung, Verwaltung und Bereitstellung Ihrer Serverzertifikate zu verwenden. In nicht unterstützten Regionen müssen Sie IAM für die Verwaltung der Zertifikate verwenden. Informationen darüber, welche Regionen ACM unterstützt, finden Sie unter AWS Certificate Manager -Endpunkte und -Kontingente im Allgemeine AWS-Referenz.

Sie können IAM-Tag-Schlüsselwertpaare verwenden, um einem Serverzertifikat benutzerdefinierte Attribute hinzuzufügen. Um beispielsweise Informationen über den Eigentümer oder Administrator eines Serverzertifikats hinzuzufügen, fügen Sie den Tag-Schlüssel **`owner`** und den Tag-Wert **`net-eng`** hinzu. Oder Sie können eine Kostenstelle angeben, indem Sie den Tag-Schlüssel **`CostCenter`** und den Tag-Wert **`1234`** hinzufügen. Sie können Tags verwenden, um den Zugriff auf Ressourcen

zu steuern oder zu kontrollieren, welche Tags mit Ressourcen verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer zusammenschließen. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Erforderliche Berechtigungen für das Markieren von Serverzertifikaten

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rolle) Serverzertifikate markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag für ein Serverzertifikat hinzuzufügen, aufzulisten oder zu entfernen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *< CertificateName >* durch den Namen des Serverzertifikats, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), einem bestimmten Serverzertifikat ein Tag hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags für ein bestimmtes Serverzertifikat hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagServerCertificate`-Aktion erfordert, dass Sie auch die `iam:ListServerCertificateTags`-Aktion einbeziehen.

Um diese Richtlinie zu verwenden, ersetzen Sie *< CertificateName >* durch den Namen des Serverzertifikats, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, FullAccess um vollen Zugriff auf IAM zu gewähren.

Verwaltung von Tags auf Serverzertifikaten (AWS CLI oder AWS APIs)

Sie können Tags für Serverzertifikate auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für Serverzertifikate zu verwalten.

Um die Tags aufzulisten, die derzeit an ein Serverzertifikat (AWS CLI oder eine AWS API) angehängt sind

- AWS CLI: als Ziel list-server-certificate-tags
- AWS API: ListServerCertificateTags

Um Tags an ein Serverzertifikat (AWS CLI oder eine AWS API) anzuhängen

- AWS CLI: war ich tag-server-certificate

- AWS API: TagServerCertificate

Um Tags aus einem Serverzertifikat (AWS CLI oder einer AWS API) zu entfernen

- AWS CLI: war ich untag-server-certificate
- AWS API: UntagServerCertificate

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Markieren virtueller MFA-Geräte

Sie können IAM-Tag-Schlüsselwertpaare verwenden, um einem virtuellen MFA-Gerät benutzerdefinierte Attribute hinzuzufügen. Um beispielsweise Kostenstelleninformationen für das virtuelle MFA-Gerät eines Benutzers hinzuzufügen, können Sie den Tag-Schlüssel **`CostCenter`** und den Tag-Wert **`1234`** hinzufügen. Sie können Tags verwenden, um den Zugriff auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einem Objekt verknüpft werden können. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Sie können Tags auch verwenden AWS STS , um benutzerdefinierte Attribute hinzuzufügen, wenn Sie eine Rolle übernehmen oder einen Benutzer zusammenschließen. Weitere Informationen finden Sie unter Sitzungs-Tags übergeben AWS STS.

Erforderliche Berechtigungen für das Markieren von virtuellen MFA-Geräten

Sie müssen Berechtigungen konfigurieren, damit eine IAM-Entität (Benutzer oder Rolle) virtuelle MFA-Geräte markieren kann. Sie können eine oder alle der folgenden IAM-Tag-Aktionen in einer IAM-Richtlinien angeben:

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`
- `iam:UntagMFADevice`

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), ein Tag für ein virtuelles MFA-Gerät hinzuzufügen, aufzulisten oder zu entfernen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags verwalten soll. Verwenden Sie Ihre Kontonummer und ersetzen Sie *<MFATokenID>* durch den Namen des virtuellen MFA-Geräts, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

So gestatten Sie einer IAM-Entität (Benutzer oder Rolle), einem bestimmten virtuellen MFA-Geräts ein Tag hinzuzufügen

Fügen Sie die folgende Anweisung der Berechtigungsrichtlinie für die IAM-Entität hinzu, die Tags für eine bestimmtes virtuelles MFA-Geräts hinzufügen, jedoch nicht entfernen soll.

> **Note**
>
> Die `iam:TagMFADevice`-Aktion erfordert, dass Sie auch die `iam:ListMFADeviceTags`-Aktion einbeziehen.

Um diese Richtlinie zu verwenden, ersetzen Sie *<MFATokenID>* durch den Namen des virtuellen MFA-Geräts, dessen Tags verwaltet werden müssen. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
```

```
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Alternativ können Sie eine AWS verwaltete Richtlinie wie IAM verwenden, FullAccess um vollen Zugriff auf IAM zu gewähren.

Verwaltung von Tags auf virtuellen MFA-Geräten (AWS CLI oder AWS API)

Sie können Tags für ein virtuelles MFA-Gerät auflisten, anfügen oder entfernen. Sie können die AWS CLI oder die AWS API verwenden, um Tags für ein virtuelles MFA-Gerät zu verwalten.

Um die Tags aufzulisten, die derzeit an ein virtuelles MFA-Gerät (AWS CLI oder eine AWS API) angehängt sind

- AWS CLI: war ich list-mfa-device-tags
- AWS API: ListMFA DeviceTags

So hängen Sie Tags an ein virtuelles MFA-Gerät (AWS CLI oder eine AWS API) an

- AWS CLI: war ich tag-mfa-device
- AWS API: TagMFADevice

So entfernen Sie Tags von einem virtuellen MFA-Gerät (AWS CLI oder einer AWS API)

- AWS CLI: war ich untag-mfa-device
- AWS API: UntagMFADevice

Informationen zum Anhängen von Tags an Ressourcen für andere AWS Dienste finden Sie in der Dokumentation zu diesen Diensten.

Weitere Informationen über die Verwendung von Tags, um präzisere Berechtigungen mit IAM-Berechtigungsrichtlinien festzulegen, finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Sitzungs-Tags übergeben AWS STS

Sitzungs-Tags sind Schlüssel-Wert-Paarattribute, die Sie übergeben, wenn Sie eine IAM-Rolle übernehmen oder in AWS STS einen Benutzer in einen Verbund aufnehmen. Sie tun dies, indem

Sie eine AWS CLI oder AWS API-Anfrage über AWS STS oder über Ihren Identitätsanbieter (IdP) stellen. Wenn Sie temporäre AWS STS Sicherheitsanmeldedaten anfordern, generieren Sie eine Sitzung. Sitzungen laufen ab und erfordern Anmeldeinformationen, z. B. ein Zugriffsschlüsselpaar und ein Sitzungstoken. Wenn Sie die Sitzungsanmeldeinformationen für eine nachfolgende Anfrage verwenden, enthält der Anforderungskontext den `aws:PrincipalTag`-Kontextschlüssel. Sie können den `aws:PrincipalTag`-Schlüssel im `Condition`-Element ihrer Richtlinien verwenden, um den Zugriff basierend auf diesen Tags zuzulassen oder abzulehnen.

Wenn Sie zur Erstellung einer Anforderung temporäre Anmeldeinformationen verwenden, enthält Ihr Auftraggeber möglicherweise eine Reihe von Tags. Diese Tags stammen aus den folgenden Quellen:

1. Sitzungs-Tags — Die Tags, die übergeben werden, wenn Sie die Rolle übernehmen oder den Benutzer mithilfe der AWS API AWS CLI oder verbinden. Weitere Informationen zu diesen Operationen finden Sie unter Sitzungs-Tagging-Operationen.
2. Eingehende transitive Sitzungs-Tags – Diese Tags wurden von einer vorherigen Sitzung in einer Rollenkette übernommen. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags an späterer Stelle in diesem Thema.
3. IAM-Tags – Die Tags, die an die von IAM übernommene Rolle angehängt sind.

Themen

Sitzungs-Tagging-Operationen

Sie können Sitzungs-Tags mithilfe der folgenden AWS CLI oder AWS API-Operationen in AWS STSübergeben. Mit der Funktion „Rolle AWS Management Console wechseln“ können Sie keine Sitzungs-Tags übergeben.

Sie können die Sitzungs-Tags auch als transitiv festlegen. Transitive Tags bleiben während der Rollenverkettung erhalten. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.

Vergleichsmethoden zum Übergeben von Sitzungs-Tags

Operation	Wer die Rolle annehmen kann	Methode zum Übergeben von Tags	Methode zum Festlegen von transitiven Tags
`assume-role` -CLI- oder `AssumeRole` -API-Operation	IAM-Benutzer oder eine Sitzung	`Tags`-API-Parameter oder `--tags`-CLI-Option	`TransitiveTagKeys` -API-Parameter oder `--transitive-tag-keys` -CLI-Option
`assume-role-with-saml` -CLI- oder `AssumeRoleWithSAML` -API-Operation	Jeder Benutzer, der über einen SAML-Identitätsanbieter authentifiziert wird	`PrincipalTag` -SAML-Attribut	`TransitiveTagKeys` -SAML-Attribut
`assume-role-with-web-identity` -CLI- oder `AssumeRoleWithWebIdentity` -API-Operation	Jeder Benutzer, der über einen OIDC-Anbieter authentifiziert wurde	`PrincipalTag` OIDC-Token	`TransitiveTagKeys` OIDC-Token

Operation	Wer die Rolle annehmen kann	Methode zum Übergeben von Tags	Methode zum Festlegen von transitiven Tags
`get-federation-token` -CLI- oder `GetFederationToken` -API-Operation	IAM-Benutzer oder Stammbenutzer	`Tags`-API-Parameter oder `--tags`-CLI-Option	Nicht unterstützt

Operationen, die die Sitzungsmarkierung unterstützen, können fehlschlagen, wenn eine der folgenden Bedingungen zutrifft:

- Sie übergeben mehr als 50 Sitzungs-Tags.
- Der Klartext Ihrer Sitzungs-Tag-Schlüssel umfasst mehr als 128 Zeichen.
- Der Klartext Ihrer Sitzungs-Tag-Werte umfasst mehr als 256 Zeichen.
- Die Gesamtgröße des Klartexts der Sitzungsrichtlinien umfasst mehr als 2.048 Zeichen.
- Die gepackte Gesamtgröße der Sitzungsrichtlinien und Sitzungs-Tags ist zu groß. Wenn die Operation fehlschlägt, gibt die Fehlermeldung anhand eines Prozentsatzes an, wie nahe die Richtlinien und Tags zusammengenommen an der oberen Größengrenze liegen.

Wissenswertes über Sitzungs-Tags

Lesen Sie vor der Verwendung von Sitzungs-Tags die folgenden Informationen zu Sitzungen und Tags.

- Bei der Verwendung von Sitzungs-Tags müssen Vertrauensrichtlinien für alle Rollen, die mit dem Identitätsanbieter (IdP) verbunden sind, der Tags übergibt, die `sts:TagSession`-Berechtigung haben. Bei Rollen, die diese Berechtigung in der Vertrauensrichtlinie nicht haben, schlägt der `AssumeRole`-Vorgang fehl.
- Wenn Sie eine Sitzung anfordern, können Sie Auftraggeber-Tags als Sitzungs-Tags angeben. Die Tags gelten für Anforderungen, die Sie mit den Anmeldeinformationen der Sitzung durchführen.

- Sitzungs-Tags sind Schlüssel-Wert-Paare. Wenn Sie beispielsweise Kontaktinformationen zu einer Sitzung hinzufügen möchten, können Sie den Sitzungs-Tag-Schlüssel `email` und den Tag-Wert `johndoe@example.com` hinzufügen.
- Sitzungs-Tags müssen den Regeln für die Benennung von Tags in IAM und entsprechen. AWS STS Dieses Thema enthält Informationen zur Unterscheidung von Groß-/Kleinschreibung und zu eingeschränkten Präfixen, die für Ihre Sitzungs-Tags gelten.
- Neue Sitzungs-Tags überschreiben vorhandene Tags für übernommene Rollen oder Verbundbenutzer mit demselben Tag-Schlüssel, unabhängig von der Groß-/Kleinschreibung.
- Sie können Sitzungs-Tags nicht mit dem AWS Management Consoleübergeben.
- Sitzungs-Tags sind nur für die aktuelle Sitzung gültig.
- Sitzungs-Tags unterstützen Rollenverkettung. Übergibt standardmäßig AWS STS keine Tags an nachfolgende Rollensitzungen. Sie können Sitzungs-Tags jedoch als transitiv festlegen. Transitive Tags bleiben während der Rollenverkettung bestehen und ersetzen übereinstimmende `ResourceTag`-Werte nach der Auswertung der Rollenvertrauensrichtlinie. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.
- Sie können Sitzungs-Tags verwenden, um den Zugriff auf Ressourcen zu steuern oder zu steuern, welche Tags an eine nachfolgende Sitzung übergeben werden können. Weitere Informationen finden Sie unter IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC.
- Sie können die Auftraggeber-Tags für Ihre Sitzung, einschließlich der zugehörigen Sitzungs-Tags, in den AWS CloudTrail -Protokollen anzeigen. Weitere Informationen finden Sie unter Sitzungs-Tags anzeigen in CloudTrail.
- Sie müssen für jedes Sitzungs-Tag einen einzelnen Wert übergeben. AWS STS unterstützt keine mehrwertigen Sitzungs-Tags.
- Sie können maximal 50 Sitzungs-Tags übergeben. Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.
- Bei einer AWS Konvertierung werden die übergebenen Sitzungsrichtlinien und Sitzungs-Tags in ein komprimiertes Binärformat mit einem separaten Limit komprimiert. Wenn Sie dieses Limit überschreiten, wird in der AWS CLI oder AWS API-Fehlermeldung angezeigt, wie nahe die kombinierten Richtlinien und Tags in Prozent an die obere Größenbeschränkung herankommen.

Erforderliche Berechtigungen zum Hinzufügen von Sitzungs-Tags

Zusätzlich zu der Aktion, die der API-Operation entspricht, muss in Ihrer Richtlinie die folgende reine Berechtigungsaktion enthalten sein:

```
sts:TagSession
```

> **Important**
>
> Bei der Verwendung von Sitzungs-Tags müssen die Rollenvertrauensrichtlinien für alle Rollen, die mit einem Identitätsanbieter (IdP) verbunden sind, über die `sts:TagSession`-Berechtigung verfügen. Der `AssumeRole`-Vorgang schlägt für jede Rolle fehl, die mit einem Identitätsanbieter verbunden ist, der Sitzungs-Tags ohne diese Berechtigung übergibt. Wenn Sie die Vertrauensrichtlinie für Rollen nicht für jede Rolle aktualisieren möchten, können Sie eine separate IdP-Instance zum Übergeben von Sitzungs-Tags verwenden. Fügen Sie dann die `sts:TagSession`-Berechtigung nur den Rollen hinzu, die mit dem separaten IdP verbunden sind.

Sie können diese `sts:TagSession`-Aktion mit den folgenden Bedingungsschlüsseln verwenden.

- `aws:PrincipalTag` – Verwenden Sie diesen Schlüssel, um das Tag, das dem Auftraggeber angefügt ist, der die Anforderung stellt, mit dem Tag zu vergleichen, das Sie in der Richtlinie angeben. Beispielsweise können Sie einem Auftraggeber nur erlauben, Sitzungs-Tags zu übergeben, wenn der Auftraggeber, der die Anforderung stellt, über die angegebenen Tags verfügt.
- `aws:RequestTag` – Verwenden Sie diesen Schlüssel, um das Tag-Schlüssel-Wert-Paar, das in der Anforderung übergeben wurde, mit dem Tag-Paar zu vergleichen, das Sie in der Richtlinie angeben. Sie können dem Auftraggeber beispielsweise gestatten, die angegebenen Sitzungs-Tags zu übergeben, jedoch nur mit den angegebenen Werten.
- `aws:ResourceTag` – Verwenden Sie diesen Schlüssel, um das Tag-Schlüssel-Wert-Paar, das Sie in der Richtlinie angeben, mit dem Schlüssel-Wert-Paar zu vergleichen, das der Ressource angefügt ist. Sie können dem Auftraggeber beispielsweise nur gestatten, Sitzungs-Tags zu übergeben, wenn die von ihnen übernommene Rolle die angegebenen Tags enthält.
- `aws:TagKeys` – Verwenden Sie diesen Schlüssel, um die Tag-Schlüssel in einer Anforderung mit den Schlüsseln zu vergleichen, die Sie in der Richtlinie angeben. Sie können dem Auftraggeber beispielsweise nur gestatten, Sitzungs-Tags mit den angegebenen Tag-Schlüsseln zu übergeben.

Dieser Bedingungsschlüssel begrenzt die maximale Gruppe von Sitzungs-Tags, die übergeben werden können.

- `sts:TransitiveTagKeys` - Verwenden Sie diesen Schlüssel, um die transitiven Sitzungs-Tag-Schlüssel in der Anforderung mit den in der Richtlinie angegebenen Schlüsseln zu vergleichen. Beispielsweise können Sie eine Richtlinie schreiben, um einem Auftraggeber nur die Festlegung bestimmter Tags als transitiv zu gestatten. Transitive Tags bleiben während der Rollenverkettung erhalten. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.

Mit der folgenden Vertrauensrichtlinie für Rollen kann der `test-session-tags`-Benutzer beispielsweise die Rolle übernehmen, der die Richtlinie angefügt ist. Wenn dieser Benutzer die Rolle übernimmt, muss er die AWS API AWS CLI oder verwenden, um die drei erforderlichen Sitzungs-Tags und die erforderliche externe ID zu übergeben. Darüber hinaus kann der Benutzer entscheiden, ob er die `Project`- und `Department`-Tags als transitiv festlegen möchte.

Example Beispiel einer Rollenvertrauensrichtlinie für Sitzungs-Tags

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
```

```
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

Was macht diese Richtlinie?

- Mit der `AllowIamUserAssumeRole`-Anweisung kann der `test-session-tags`-Benutzer die Rolle übernehmen, der die Richtlinie angefügt ist. Wenn dieser Benutzer die Rolle übernimmt, muss er die erforderlichen Sitzungs-Tags und die externe ID übergeben.
  - Der erste Bedingungsblock dieser Anweisung verlangt, dass der Benutzer die Sitzungs-Tags `Project`, `CostCenter` und `Department` übergibt. Die Tag-Werte spielen in dieser Anweisung keine Rolle, daher haben wir Platzhalter (*) für die Tag-Werte verwendet. Dieser Block stellt sicher, dass der Benutzer mindestens diese drei Sitzungs-Tags übergibt, andernfalls schlägt die Operation fehl. Andernfalls schlägt die Operation fehl. Der Benutzer kann zusätzliche Tags übergeben.
  - Der zweite Bedingungsblock verlangt, dass der Benutzer eine externe ID mit dem Wert `Example987` übergibt.
- Die `AllowPassSessionTagsAndTransitive`-Anweisung lässt die reine Berechtigungsaktion `sts:TagSession` zu. Diese Aktion muss zulässig sein, damit der Benutzer Sitzungs-Tags übergeben kann. Wenn Ihre Richtlinie die erste, nicht jedoch die zweite Anweisung enthält, kann der Benutzer die Rolle nicht übernehmen.
  - Der erste Bedingungsblock dieser Anweisung ermöglicht es dem Benutzer, einen beliebigen Wert für die Sitzungs-Tags `CostCenter` und `Projectzu` übergeben. Dazu verwenden Sie

Platzhalter (*) für den Tag-Wert in der Richtlinie, wofür Sie den StringLikeBedingungsoperator verwenden müssen.

- Der zweite Bedingungsblock ermöglicht es dem Benutzer, nur den`Engineering`- oder `Marketing`-Wert oder für das Sitzungs-Tag `Department` zu übergeben.
- Der dritte Bedingungsblock listet die maximale Gruppe von Tags auf, die als transitiv festgelegt werden können. Der Benutzer hat die Möglichkeit, eine Teilmenge oder keine Tags als transitiv festzulegen. Es ist jedoch nicht möglich, zusätzliche Tags als transitiv festzulegen. Sie können verlangen, dass mindestens eines der Tags als transitiv festgelegt wird. Zu diesem Zweck müssen Sie einen weiteren Bedingungsblock mit der Angabe `"Null": {"sts:TransitiveTagKeys":"false"}` hinzufügen.

Übergeben von Sitzungs-Tags mit AssumeRole

Der `AssumeRole` Vorgang gibt einen Satz temporärer Anmeldeinformationen zurück, mit denen Sie auf AWS Ressourcen zugreifen können. Sie können IAM-Benutzer- oder -Rollenanmeldeinformationen zum Aufrufen von `AssumeRole` verwenden. Verwenden Sie die `--tags` AWS CLI Option oder den `Tags` AWS API-Parameter, um Sitzungs-Tags zu übergeben, während Sie eine Rolle übernehmen.

Um Tags als transitiv festzulegen, verwenden Sie die `--transitive-tag-keys` AWS CLI Option oder den `TransitiveTagKeys` AWS API-Parameter. Transitive Tags bleiben während der Rollenverkettung erhalten. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.

Das folgende Beispiel zeigt eine Beispielanforderung, bei der `AssumeRole` verwendet wird. Wenn Sie in diesem Beispiel die Rolle `my-role-example` übernehmen, erstellen Sie eine Sitzung mit dem Namen `my-session`. Sie fügen die Sitzungs-Tag-Schlüssel-Wert-Paare `Project` = `Automation`, `CostCenter` = `12345` und `Department` = `Engineering` hinzu. Darüber hinaus legen Sie die `Project`- und `Department`-Tags als transitiv fest, indem Sie ihre Schlüssel angeben.

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
```

```
--external-id Example987
```

Übergabe von Sitzungs-Tags mit AssumeRoleWith SAML

Die `AssumeRoleWithSAML`-Operation wird mit dem SAML-basierten Verbund authentifiziert. Dieser Vorgang gibt einen Satz temporärer Anmeldeinformationen zurück, mit denen Sie auf AWS Ressourcen zugreifen können. Weitere Informationen zur Verwendung eines SAML-basierten Verbunds für den AWS Management Console Zugriff finden Sie unter. Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console Einzelheiten zum AWS CLI AWS API-Zugriff finden Sie unter. SAML 2.0-Verbund Ein Tutorial zur Konfiguration des SAML-Verbunds für Ihre Active Directory-Benutzer finden Sie unter AWS Verbundauthentifizierung mit Active Directory Federation Services (ADFS) im AWS Sicherheitsblog.

Als Administrator können Sie es Mitgliedern Ihres Unternehmensverzeichnisses ermöglichen, den Vorgang gemeinsam zu AWS nutzen. AWS STS `AssumeRoleWithSAML` Hierfür müssen Sie die folgenden Aufgaben ausführen:

1. Konfigurieren des Netzwerks als SAML-Anbieter für AWS
2. Erstellen eines SAML-Anbieters in IAM
3. Konfigurieren einer Rolle und ihrer Berechtigungen in AWS für Ihre Verbundbenutzer
4. Abschließen der Konfiguration des SAML-Identitätsanbieters und Erstellen von Zusicherungen für die SAML-Authentifizierungsantwort.

AWS schließt Identitätsanbieter mit zertifizierter end-to-end Erfahrung im Bereich Sitzungs-Tags mit ihren Identitätslösungen ein. Weitere Informationen zur Verwendung dieser Identitätsanbieter zum Konfigurieren von Sitzungs-Tags finden Sie unter Integrieren Sie SAML-Lösungsanbieter von Drittanbietern mit AWS.

Um SAML-Attribute als Sitzungs-Tags zu übergeben, schließen Sie das `Attribute`-Element ein und legen Sie dabei für das `Name`-Attribut `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`* fest. Verwenden Sie das `AttributeValue`-Element, um den Wert des Tags anzugeben. Fügen Sie ein separates `Attribute`-Element für jedes Sitzungs-Tag ein.

Angenommen, Sie möchten die folgenden Identitätsattribute als Sitzungs-Tags übergeben:

- `Project:Automation`
- `CostCenter:12345`
- `Department:Engineering`

Um diese Attribute zu übergeben, schließen Sie die folgenden Elemente in Ihre SAML-Zusicherung ein.

Example Beispielausschnitt einer SAML-Zusicherung

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

Um die oben genannten Tags als transitiv festzulegen, schließen Sie ein weiteres `Attribute`-Element ein und legen dabei für das Attribut `Name` den Wert`https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys` fest. Transitive Tags bleiben während der Rollenverkettung erhalten. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.

Verwenden Sie das folgende Mehrwertattribut, um die Tags `Project` und `Department` als transitiv festzulegen:

Example Beispielausschnitt einer SAML-Zusicherung

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Weitergabe von Sitzungs-Tags mit AssumeRoleWithWebIdentity

Verwenden Sie einen OpenID Connect (OIDC) -kompatiblen Verbund, um den Vorgang zu authentifizieren. `AssumeRoleWithWebIdentity` Dieser Vorgang gibt einen Satz temporärer Anmeldeinformationen zurück, mit denen Sie auf Ressourcen zugreifen können. AWS Weitere Informationen zur Verwendung des Web-Identitätsverbunds für den AWS Management Console Zugriff finden Sie unterOIDC-Föderation.

Um Sitzungs-Tags von OpenID Connect (OIDC) zu übergeben, müssen Sie die Sitzungs-Tags in das JSON Web Token (JWT) einbeziehen. Fügen Sie Sitzungs-Tags in den

`https://aws.amazon.com/tags`-Namespace in dem Token ein, wenn Sie die `AssumeRoleWithWebIdentity`-Anforderung senden. Weitere Informationen zu OIDC-Token und Ansprüchen finden Sie unter Verwenden von Token mit Benutzerpools im Amazon Cognito Developer Guide.

Bei dem folgenden dekodierten JWT handelt es sich beispielsweise um ein Token, das zum Aufrufen von `AssumeRoleWithWebIdentity` mit den Sitzungs-Tags `Project`, `CostCenter` und `Department` verwendet wird. Das Token legt zudem die Tags `Project` und `CostCenter` als transitiv fest. Transitive Tags bleiben während der Rollenverkettung erhalten. Weitere Informationen finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.

Example Beispiel für ein dekodiertes JSON-Web-Token

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

Übergabe von Sitzungs-Tags mit GetFederationToken

Mit `GetFederationToken` können Sie Ihren Benutzer in einen Verbund einbeziehen. Dieser Vorgang gibt einen Satz temporärer Anmeldeinformationen zurück, mit denen Sie auf AWS Ressourcen zugreifen können. Verwenden Sie die `--tags` AWS CLI Option oder den `Tags` AWS API-Parameter, um Ihrer Verbundbenutzersitzung Tags hinzuzufügen. Bei der Verwendung von `GetFederationToken` können Sie Sitzungs-Tags nicht als transitiv festlegen, da Sie die

temporären Anmeldeinformationen nicht verwenden können, um eine Rolle zu übernehmen. Sie können in diesem Fall keine Rollenverkettung verwenden.

Das folgende Beispiel zeigt eine Beispielanforderung unter Verwendung von `GetFederationToken`. Wenn Sie in diesem Beispiel das Token anfordern, erstellen Sie eine Sitzung mit dem Namen `my-fed-user`. Sie fügen die Sitzungs-Tag-Schlüssel-Wert-Paare `Project` = `Automation` und `Department` = `Engineering` hinzu.

Example Beispiel für eine GetFederationToken CLI-Anfrage

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Wenn Sie die temporären Anmeldeinformationen verwenden, die von der `GetFederationToken`-Operation zurückgegeben werden, enthalten die Auftraggeber-Tags der Sitzung die Tags des Benutzers und die übergebenen Sitzungs-Tags.

Verkettung von Rollen mit Sitzungs-Tags

Sie können eine Rolle übernehmen und dann die temporären Anmeldeinformationen verwenden, um eine andere Rolle zu übernehmen. Sie können von Sitzung zu Sitzung fortfahren. Dies wird als Rollenverkettung bezeichnet. Wenn Sie Sitzungs-Tags übergeben, während Sie eine Rolle übernehmen, können Sie die Schlüssel als transitiv festlegen. Dadurch wird sichergestellt, dass diese Sitzungs-Tags an nachfolgende Sitzungen in einer Rollenkette übergeben werden. Rollen-Tags können nicht als transitiv festgelegt werden. Um diese Tags an nachfolgende Sitzungen zu übergeben, müssen Sie sie als Sitzungs-Tags angeben.

> **Note**
>
> Transitive Tags bleiben während der Rollenverkettung bestehen und ersetzen übereinstimmende `ResourceTag`-Werte nach der Auswertung der Rollenvertrauensrichtlinie.

Das folgende Beispiel zeigt, wie AWS STS Sitzungs-Tags, transitive Tags und Rollentags an nachfolgende Sitzungen in einer Rollenkette weitergegeben werden.

In diesem Beispielszenario zur Rollenverkettung verwenden Sie einen IAM-Benutzerzugriffsschlüssel, AWS CLI um eine Rolle mit dem Namen anzunehmen. `Role1` Dann verwenden Sie die

resultierenden Sitzungsanmeldeinformationen, um eine zweite Rolle namens `Role2` zu übernehmen. Anschließend können Sie die Anmeldeinformationen der zweiten Sitzung verwenden, um eine dritte Rolle namens `Role3` zu übernehmen. Diese Anforderungen erfolgen als drei separate Operationen. Jede Rolle ist bereits in IAM markiert. Während jeder Anforderung übergeben Sie zusätzliche Sitzungs-Tags.

Durch die Verkettung von Rollen können Sie sicherstellen, dass Tags aus einer früheren Sitzung in den späteren Sitzungen bestehen bleiben. Wenn Sie hierfür den CLI-Befehl `assume-role` verwenden möchten, müssen Sie das Tag als Sitzungs-Tag übergeben und als transitiv festlegen. Sie übergeben das Tag `Star = 1` als Sitzungs-Tag. Das Tag `Heart = 1` ist an die Rolle angefügt und wird als Auftraggeber-Tag angewendet, wenn Sie die Sitzung verwenden. Sie möchten jedoch auch, dass das Tag `Heart = 1` automatisch an die zweite oder dritte Sitzung übergeben wird. Zu diesem Zweck fügen Sie es manuell als Sitzungs-Tag ein. Die resultierenden Sitzungsauftraggeber-Tags enthalten beide Tags und legen sie als transitiv fest.

Sie führen diese Anfrage mit dem folgenden Befehl aus: AWS CLI

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

Dann verwenden Sie die Anmeldeinformationen für diese Sitzung, um `Role2` zu übernehmen. Das Tag `Sun = 2` ist an die zweite Rolle angefügt und gilt als Auftraggeber-Tag, wenn Sie die zweite Sitzung verwenden. Die Tags `Heart` und `Star` werden von den transitiven Sitzungs-Tags in der ersten Sitzung übernommen. Die resultierenden Auftraggeber-Tags der zweiten Sitzung sind `Heart = 1`, `Star = 1` und `Sun = 2`. `Heart` und `Star` bleiben weiterhin transitiv. Das Tag `Sun`, das an `Role2` angefügt wurde, ist nicht als transitiv markiert, da es sich nicht um ein Sitzungs-Tag handelt. Dieses Tag wird nicht von zukünftigen Sitzungen übernommen.

Sie führen diese zweite Anfrage mit dem folgenden AWS CLI Befehl aus:

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

Dann verwenden Sie die Anmeldeinformationen für die zweite Sitzung, um `Role3` zu übernehmen. Die Auftraggeber-Tags für die dritte Sitzung stammen aus neuen Sitzungs-Tags, den übernommenen transitiven Sitzungs-Tags und den Rollen-Tags. Die Tags `Heart = 1` und `Star = 1` in der zweiten Sitzung wurden von dem transitiven Sitzungs-Tag in der ersten Sitzung übernommen. Wenn Sie versuchen, das Sitzungs-Tag `Sun = 2` zu übergeben, schlägt die Operation fehl. Das geerbte Sitzungs-Tag `Star = 1` hat Vorrang vor dem Tag role `Star = 3`. Bei der Rollenverkettung überschreibt der Wert eines transitiven Tags die Rolle, die dem `ResourceTag`-Wert nach der Bewertung der Rollenvertrauensrichtlinie. In diesem Beispiel verwendet `Role3` `Star` als `ResourceTag` in der Vertrauensrichtlinie der Rolle und setzt den Wert `ResourceTag` auf den Wert des transitiven Tags der aufrufenden Rollensitzung. Das Tag `Lightning` der Rolle gilt auch für die dritte Sitzung und ist nicht als transitiv festgelegt.

Sie führen die dritte Anfrage mit dem folgenden AWS CLI Befehl aus:

Example Beispiel für eine AssumeRole CLI-Anfrage

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Verwenden von Sitzungs-Tags für ABAC

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Tag-Attributen definiert werden.

Wenn Ihr Unternehmen einen SAML-basierten Identitätsanbieter (IdP) für Unternehmens-Benutzeridentitäten verwendet, können Sie Ihre SAML-Zusicherung so konfigurieren, dass Sitzungs-Tags an AWSübergeben werden. Bei Benutzeridentitäten in Unternehmen werden beispielsweise, wenn sich Ihre Mitarbeiter zusammenschließen AWS, ihre Attribute auf den resultierenden Prinzipal AWS angewendet. Sie können dann ABAC verwenden, um Berechtigungen basierend auf diesen Attributen zuzulassen oder abzulehnen. Details hierzu finden Sie unter IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC.

Weitere Informationen zur Verwendung von IAM Identity Center mit ABAC finden Sie unter Attribute für die Zugriffssteuerung im AWS IAM Identity Center -Leitfaden.

Sitzungs-Tags anzeigen in CloudTrail

Sie können AWS CloudTrail damit die Anfragen anzeigen, die zur Übernahme von Rollen oder zum Zusammenführen von Benutzern verwendet wurden. Die CloudTrail Protokolldatei

enthält Informationen über die wichtigsten Tags für die Sitzung mit übernommener Rolle oder Verbundbenutzer. Weitere Informationen finden Sie unter Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail.

Nehmen wir beispielsweise an, dass Sie eine AWS STS `AssumeRoleWithSAML` Anfrage stellen, Sitzungs-Tags übergeben und diese Tags als transitiv festlegen. Sie finden die folgenden Informationen in Ihrem CloudTrail Protokoll.

Example Beispiel für ein AssumeRoleWith SAML-Protokoll CloudTrail

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

Sie können sich die folgenden CloudTrail Beispielprotokolle ansehen, um Ereignisse anzuzeigen, die Sitzungs-Tags verwenden.

- Beispiel für ein API-Ereignis zur AWS STS Rollenverkettung in einer Protokolldatei CloudTrail
- Beispiel für ein AWS STS SAML-API-Ereignis in der Protokolldatei CloudTrail
- Beispiel für ein AWS STS OIDC-API-Ereignis in der Protokolldatei CloudTrail

Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail

IAM und AWS STS sind in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der von einem IAM-Benutzer oder einer IAM-Rolle ausgeführten Aktionen bereitstellt. CloudTrail erfasst alle API-Aufrufe für IAM und AWS STS als Ereignisse, einschließlich Aufrufe von der

Konsole und von API-Aufrufen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf einsehen. Sie können CloudTrail damit Informationen zu der Anfrage abrufen, die an IAM gestellt wurde oder AWS STS. So können Sie beispielsweise die IP-Adresse, von der aus die Anforderung gestellt wurde, wer die Anforderung gestellt hat, wann sie gestellt wurde und weitere Details einsehen.

Weitere Informationen CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

Themen

IAM und AWS STS Informationen in CloudTrail

CloudTrail ist auf Ihrem aktiviert AWS-Konto , wenn Sie das Konto erstellen. Wenn eine Aktivität in IAM oder stattfindet AWS STS, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS-Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, einschließlich Ereignissen für IAM AWS STS, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie ein Trail in der Konsole anlegen, gilt dieser 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 IAM und alle AWS STS Aktionen werden von der IAM-API-Referenz CloudTrail und der API-Referenz protokolliert und sind in diesen dokumentiert.AWS Security Token Service

Protokollierung von IAM- und API-Anfragen AWS STS

CloudTrail protokolliert alle authentifizierten API-Anfragen für IAM- und AWS STS API-Operationen. CloudTrail protokolliert auch nicht authentifizierte Anfragen zu den AWS STS Aktionen `AssumeRoleWithSAML` und`AssumeRoleWithWebIdentity`, und protokolliert die vom Identitätsanbieter bereitgestellten Informationen. Einige nicht authentifizierte AWS STS Anfragen werden jedoch möglicherweise nicht protokolliert, da sie nicht die Mindestanforderung erfüllen, dass sie ausreichend gültig sind, um als legitime Anfrage vertrauenswürdig zu sein.

Sie können die protokollierten Informationen verwenden, um Anrufe, die von einem Verbundbenutzer mit einer angenommenen Rolle getätigt wurden, dem ursprünglichen externen Verbundaufrufer zuzuordnen. Im Fall von `AssumeRole` können Sie Anrufe dem ursprünglichen AWS Dienst oder dem Konto des ursprünglichen Benutzers zuordnen. Der `userIdentity` Abschnitt der JSON-Daten im CloudTrail Protokolleintrag enthält die Informationen, die Sie benötigen, um die AssumeRole* Anfrage einem bestimmten Verbundbenutzer zuzuordnen. Weitere Informationen finden Sie unter CloudTrail UserIdentity Element im AWS CloudTrail Benutzerhandbuch.

Beispielsweise werden alle Aufrufe von IAM`CreateUser`, `DeleteRoleListGroups`, und anderen API-Vorgängen protokolliert. CloudTrail

Beispiele für solche Protokolleinträge finden Sie weiter hinten in diesem Thema.

Protokollieren von API-Anforderungen an andere AWS -Services

Authentifizierte Anfragen an andere AWS Service-API-Operationen werden von protokolliert CloudTrail, und diese Protokolleinträge enthalten Informationen darüber, wer die Anfrage generiert hat.

Angenommen, Sie haben eine Anforderung gestellt, um Amazon Amazon EC2-Instances aufzulisten oder eine AWS CodeDeploy -Bereitstellungsgruppe zu erstellen. Details über die Person oder Dienstleistung, die die Anforderung gestellt hat, sind im Protokolleintrag dieser Anforderung enthalten. Anhand dieser Informationen können Sie feststellen, ob die Anfrage von dem Root-Benutzer des AWS-Kontos, einem IAM-Benutzer, einer Rolle oder einem anderen AWS Dienst gestellt wurde.

Weitere Informationen zu den Benutzeridentitätsinformationen in CloudTrail Protokolleinträgen finden Sie unter UserIdentity Element im AWS CloudTrail Benutzerhandbuch.

Protokollieren von Benutzeranmeldeereignissen

CloudTrail protokolliert Anmeldeereignisse in den AWS Management Console, den AWS Diskussionsforen und. AWS Marketplace CloudTrailprotokolliert erfolgreiche und fehlgeschlagene Anmeldeversuche für IAM-Benutzer und Verbundbenutzer.

CloudTrail Beispielereignisse für erfolgreiche und erfolglose Root-Benutzeranmeldungen finden Sie im Benutzerhandbuch unter Beispielereignisdatensätze für Root-Benutzer.AWS CloudTrail

Aus Sicherheitsgründen wird der eingegebene Text des IAM-Benutzernamens AWS nicht protokolliert, wenn der Anmeldefehler auf einen falschen Benutzernamen zurückzuführen ist. Der Benutzername wird durch den Wert `HIDDEN_DUE_TO_SECURITY_REASONS` maskiert. Ein Beispiel hierzu finden Sie unter Beispiel für eine Anmeldung, die aufgrund eines falschen Benutzernamens fehlgeschlagen ist. an späterer Stelle in diesem Thema. Der Benutzername ist verdeckt, da solche Fehler oftmals von Benutzern begangen werden. Durch die Protokollierung dieser Fehler könnten potenziell sensible Informationen offengelegt werden. Zum Beispiel:

- Sie haben versehentlich Ihr Passwort in das Feld Benutzername eingegeben.
- Sie wählen den Link für die Anmeldeseite eines Benutzers aus AWS-Konto, geben dann aber die Kontonummer für einen anderen ein. AWS-Konto
- Ein Benutzer hat vergessen, bei welchem Konto er sich gerade anmeldet, und gibt versehentlich den Kontonamen seines privaten E-Mail-Kontos, seine Bankkontonummer oder eine andere private ID ein.

Protokollieren von Anmeldeereignissen bei temporären Anmeldeinformationen

Wenn ein Principal temporäre Anmeldeinformationen anfordert, bestimmt der Prinzipaltyp, wie das Ereignis CloudTrail protokolliert wird. Dies kann kompliziert sein, wenn ein Auftraggeber eine Rolle in einem anderen Konto annimmt. Es gibt mehrere API-Aufrufe, durch die Operationen im Zusammenhang mit kontoübergreifenden Rollenoperationen durchgeführt werden. Zunächst ruft der Principal eine AWS STS API auf, um die temporären Anmeldeinformationen abzurufen. Dieser Vorgang wird im aufrufenden Konto und in dem Konto, in dem der AWS STS Vorgang ausgeführt wird, protokolliert. Anschließend verwendet der Auftraggeber die Rolle, um andere API-Aufrufe im Konto der übernommenen Rolle auszuführen.

Sie können den `sts:SourceIdentity`-Bedingungsschlüssel in der Rollenvertrauensrichtlinie verwenden, damit Benutzer einen Sitzungsnamen angeben müssen, wenn sie eine Rolle übernehmen. Sie können beispielsweise verlangen, dass IAM-Benutzer ihren eigenen Benutzernamen als Sitzungsnamen angeben. Auf diese Weise können Sie feststellen, welcher Benutzer eine bestimmte Aktion in AWS ausgeführt hat. Weitere Informationen finden Sie unter `sts:SourceIdentity`. Sie können auch `sts:RoleSessionName` verwenden, um von den Benutzern zu verlangen, dass sie einen Sitzungsnamen angeben, wenn sie eine Rolle übernehmen. Auf diese Weise können Sie bei der Überprüfung der AWS CloudTrail Protokolle zwischen Rollensitzungen für eine Rolle unterscheiden, die von verschiedenen Hauptbenutzern verwendet wird.

Die folgende Tabelle zeigt, wie verschiedene Benutzeridentitätsinformationen für jede der AWS STS APIs CloudTrail protokolliert werden, die temporäre Anmeldeinformationen generieren.

Auftraggebertyp	STS-API	Benutzeridentität im CloudTrail Protokoll für das Konto des Anrufers	Benutzeridentität im CloudTrail Protokoll für das Konto der angenommenen Rolle	Benutzeridentität im CloudTrail Protokoll für die nachfolgenden API-Aufrufe der Rolle
Root-Benutzer des AWS-Kontos Anmeldeinformationen	GetSessionToken	Stammbenutzeridentität	Die Rolle beinhaltendes Konto ist mit	Stammbenutzeridentität

Auftraggebertyp	STS-API	Benutzeridentität im CloudTrail Protokoll für das Konto des Anrufers	Benutzeridentität im CloudTrail Protokoll für das Konto der angenommenen Rolle	Benutzeridentität im CloudTrail Protokoll für die nachfolgenden API-Aufrufe der Rolle
			aufrufendem Konto identisch	
IAM-Benutzer	GetSessionToken	IAM-Benutzeridentität	Die Rolle beinhaltendes Konto ist mit aufrufendem Konto identisch	IAM-Benutzeridentität
IAM-Benutzer	GetFederationToken	IAM-Benutzeridentität	Die Rolle beinhaltendes Konto ist mit aufrufendem Konto identisch	IAM-Benutzeridentität
IAM-Benutzer	AssumeRole	IAM-Benutzeridentität	Kontonummer und Prinzipal-ID (falls es sich um einen Benutzer handelt) oder AWS Dienstprinzipal	Nur Rollenidentität (kein Benutzer)
Extern authentifizierter Benutzer	AssumeRoleWithSAML	–	SAML-Benutzeridentität	Nur Rollenidentität (kein Benutzer)
Extern authentifizierter Benutzer	AssumeRoleWithWebIdentity	–	OIDC/Web-Benutzeridentität	Nur Rollenidentität (kein Benutzer)

CloudTrail betrachtet eine Aktion als schreibgeschützt, wenn sie keine mutierende Wirkung auf eine Ressource hat. Beim Protokollieren eines schreibgeschützten Ereignisses werden die Informationen im Protokoll CloudTrail geschwärzt. `responseElements` Wenn ein Ereignis CloudTrail protokolliert wird, das nicht schreibgeschützt ist, wird die vollständige Information im `responseElements` Protokolleintrag angezeigt. Bei den AWS STS APIs CloudTrail wird jedoch `AssumeRole` `AssumeRoleWithSAML`AssumeRoleWithWebIdentity, obwohl sie schreibgeschützt protokolliert wurden, die vollständige Information `responseElements` in das Protokoll für diese APIs aufgenommen.

Die folgende Tabelle zeigt, wie CloudTrail Protokolle `responseElements` und `readOnly` Informationen für jede der AWS STS APIs temporäre Anmeldeinformationen generieren.

STS-API	Informationen zu den Antwortelementen	Read-only
AssumeRole	Enthalten	true
AssumeRoleWithSAML	Inbegriffen	true
AssumeRoleWithWebIdentity	Inbegriffen	true
GetFederationToken	Inbegriffen	false
GetSessionToken	Inbegriffen	false

Beispiel für IAM-API-Ereignisse im Protokoll CloudTrail

CloudTrail Protokolldateien enthalten Ereignisse, die mit JSON formatiert wurden. Ein API-Ereignis stellt eine einzelne API-Anforderung dar und enthält Informationen zum Auftraggeber, der angeforderten Aktion, etwaigen Parametern und dem Datum und der Uhrzeit der Aktion.

Beispiel für ein IAM-API-Ereignis in einer Protokolldatei CloudTrail

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für eine Anfrage für die IAM-Aktion`GetUserPolicy`.

```
{
  "eventVersion": "1.05",
```

```
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/JaneDoe",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "userName": "JaneDoe",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-07-15T21:39:40Z"
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
  "eventTime": "2014-07-15T21:40:14Z",
  "eventSource": "iam.amazonaws.com",
  "eventName": "GetUserPolicy",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "signin.amazonaws.com",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "userName": "JaneDoe",
    "policyName": "ReadOnlyAccess-JaneDoe-201407151307"
  },
  "responseElements": null,
  "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
  "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"
}
```


Diesen Ereignisinformationen können Sie im Element `ReadOnlyAccess-JaneDoe-201407151307` entnehmen, dass hier die Benutzerrichtlinie `JaneDoe` für den Benutzer `requestParameters` angefordert wurde. Außerdem können Sie sehen, dass die Anforderung von dem IAM-Benutzer `JaneDoe` am 15. Juli 2014 um 21:40 Uhr (UTC) gemacht wurde. In diesem Fall stammt die Anfrage von AWS Management Console, wie Sie dem `userAgent` Element entnehmen können.

Beispiel für AWS STS API-Ereignisse im CloudTrail Protokoll

CloudTrail Protokolldateien enthalten Ereignisse, die mit JSON formatiert sind. Ein API-Ereignis stellt eine einzelne API-Anforderung dar und enthält Informationen zum Auftraggeber, der angeforderten Aktion, etwaigen Parametern und dem Datum und der Uhrzeit der Aktion.

Beispiel für kontoübergreifende AWS STS API-Ereignisse in Protokolldateien CloudTrail

Der `JohnDoe` im Konto 777788889999 genannte IAM-Benutzer ruft die AWS STS `AssumeRole` Aktion auf, um die Rolle im Konto 111122223333 anzunehmen. `EC2-dev` Der Kontoadministrator verlangt, dass Benutzer eine Quellidentität festlegen, die ihrem Benutzernamen entspricht, wenn sie die Rolle übernehmen. Der Benutzer gibt den Wert der Quellidentität von `JohnDoe`.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/JohnDoe",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
  "sourceIdentity": "JohnDoe"
  },
```

```
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Das zweite Beispiel zeigt den Protokolleintrag des angenommenen Rollenkontos (111122223333) für dieselbe Anfrage. CloudTrail

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
    },
```

```
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
      },
  "sourceIdentity": "JohnDoe"
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

Beispiel für ein API-Ereignis zur AWS STS Rollenverkettung in einer Protokolldatei CloudTrail

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für eine Anfrage von John Doe im Konto 111111111111. John verwendete zuvor seinen `JohnDoe`-Benutzer, um die `JohnRole1`-Rolle zu übernehmen. Für diese Anforderung verwendet er die Anmeldeinformationen dieser Rolle, um die `JohnRole2`-Rolle zu übernehmen. Dies wird als Rollenverkettung bezeichnet. Die Quellidentität, die er bei der Übernahme der `JohnDoe1`-Rolle festgelegt hat, bleibt bei der Anforderung, `JohnRole2` zu übernehmen, bestehen. Wenn John versucht, bei der Übernahme der Rolle eine andere Quellidentität festzulegen, wird die Anforderung abgelehnt. John übergibt zwei Sitzungs-Tags in die Anforderung. Er setzt diese beiden Tags als transitiv fest. Die Anforderung übernimmt das `Department`-Tag als transitiv, weil John es als transitiv festgelegt hat, als `JohnRole1` übernommen hat. Weitere Informationen zu Quellidentität finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen. Weitere Hinweise zu transitiven Schlüsseln in Rollenketten finden Sie unter Verkettung von Rollen mit Sitzungs-Tags.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/JohnDoe/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
                "sessionIssuer": {
```

```
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "JohnDoe"
            },
            "sourceIdentity": "JohnDoe"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "JohnDoe",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
            "expiration": "Oct 2, 2019, 11:12:29 PM",
```

```
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
    "sourceIdentity": "JohnDoe"
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```


Beispiel für ein AWSAWS STS Service-API-Ereignis in einer Protokolldatei CloudTrail

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für eine Anfrage, die von einem AWS Dienst gestellt wurde, der mithilfe von Berechtigungen einer Servicerolle eine andere Service-API aufruft. Es zeigt den CloudTrail Protokolleintrag für die Anfrage, die im Konto 777788889999 gestellt wurde.


```
{
  "eventVersion": "1.04",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
```

```
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "my-test-bucket-cross-account"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```


Beispiel für ein AWS STS SAML-API-Ereignis in der Protokolldatei CloudTrail

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für eine Anforderung, die für die AWS STS `AssumeRoleWithSAML` Aktion gestellt wurde. Die Anforderung enthält die SAML-Attribute `CostCenter` und `Project`, die durch die SAML-Assertion als Sitzungs-Tags übergeben werden. Diese Tags werden als transitiv festgelegt, so dass sie in Rollenverkettungsszenarien bestehen bleiben. Die Anfrage enthält den optionalen API-Parameter`DurationSeconds`, der wie `durationSeconds` im CloudTrail Protokoll dargestellt wird, und ist auf `1800` Sekunden festgelegt.

Die Anforderung enthält außerdem das SAML-Attribut `sourceIdentity`, das in der SAML-Assertion übergeben wird. Wenn jemand die resultierenden Anmeldeinformationen für die Rollensitzung verwendet, um eine andere Rolle zu übernehmen, bleibt diese Quellidentität bestehen.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
        },
```

```
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 1,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```

Beispiel für ein AWS STS OIDC-API-Ereignis in der Protokolldatei CloudTrail

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für eine Anforderung, die für die AWS STS `AssumeRoleWithWebIdentity` Aktion gestellt wurde. Die Anforderung enthält die Attribute

`CostCenter` und `Project`, die durch das Identitätsanbietertoken als Sitzungs-Tags übergeben werden. Diese Tags werden als transitiv festgelegt, so dass sie in Rollenverkettungsszenarien bestehen bleiben. Die Anforderung enthält das `sourceIdentity`-Attribut aus dem Token des Identitätsanbieters. Wenn jemand die resultierenden Anmeldeinformationen für die Rollensitzung verwendet, um eine andere Rolle zu übernehmen, bleibt diese Quellidentität bestehen.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "accounts.google.com:<id-of-
application>.apps.googleusercontent.com:<id-of-user>",
    "userName": "<id of user>",
    "identityProvider": "accounts.google.com"
  },
  "eventTime": "2016-03-23T01:39:51Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5",
  "requestParameters": {
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "MyAssignedRoleSessionName"
        "principalTags": {
            "CostCenter": "24680",
            "Project": "Pegasus"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
  },
  "responseElements": {
    "provider": "accounts.google.com",
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "audience": "<id of application>.apps.googleusercontent.com",
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "expiration": "Mar 23, 2016, 2:39:51 AM",
```

```
      "sessionToken": "<encoded session token blob>"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:MyAssignedRoleSessionName",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/
MyAssignedRoleSessionName"
    }
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
      "accountId": "444455556666",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
  "eventID": "bEXAMPLE-0b30-4246-b28c-e3da3EXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "444455556666"
}
```

Beispiel für Anmeldeereignisse im CloudTrail-Protokoll

CloudTrail Protokolldateien enthalten Ereignisse, die mit JSON formatiert sind. Ein Anmeldeereignis stellt eine einzelne Anmeldeanforderung dar und enthält Informationen über den Anmelde-Auftraggeber, die Region sowie das Datum und die Uhrzeit der Aktion.

Beispiel für ein erfolgreiches Anmeldeereignis in der Protokolldatei CloudTrail

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für ein erfolgreiches Anmeldeereignis.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JohnDoe",
    "accountId": "111122223333",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
```

```
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

Weitere Informationen zu den in CloudTrail Protokolldateien enthaltenen Informationen finden Sie unter CloudTrail Event Reference im AWS CloudTrail Benutzerhandbuch.

Beispiel für ein fehlgeschlagenes Anmeldeereignis in der CloudTrail Protokolldatei

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für ein fehlgeschlagenes Anmeldeereignis.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
```

```
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

Anhand dieser Informationen können Sie feststellen, dass der Anmeldeversuch von einer IAM-Benutzerin namens `JaneDoe` unternommen wurde, wie auch im `userIdentity`-Element zu sehen ist. Außerdem können Sie dem Element `responseElements` entnehmen, dass die Anmeldung fehlgeschlagen ist. Den Informationen zufolge hat `JaneDoe` am 8. Juli 2014 um 17:35 Uhr (UTC) versucht, sich bei der Amazon SNS-Konsole anzumelden.

Beispiel für eine Anmeldung, die aufgrund eines falschen Benutzernamens fehlgeschlagen ist.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag für ein fehlgeschlagenes Anmeldeereignis, das dadurch verursacht wurde, dass der Benutzer einen falschen Benutzernamen eingegeben hat. AWS maskiert den `userName` Text mit`HIDDEN_DUE_TO_SECURITY_REASONS`, um zu verhindern, dass potenziell vertrauliche Informationen preisgegeben werden.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
```

```
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```

IAM-Rolle, Vertrauensrichtlinie, Verhalten

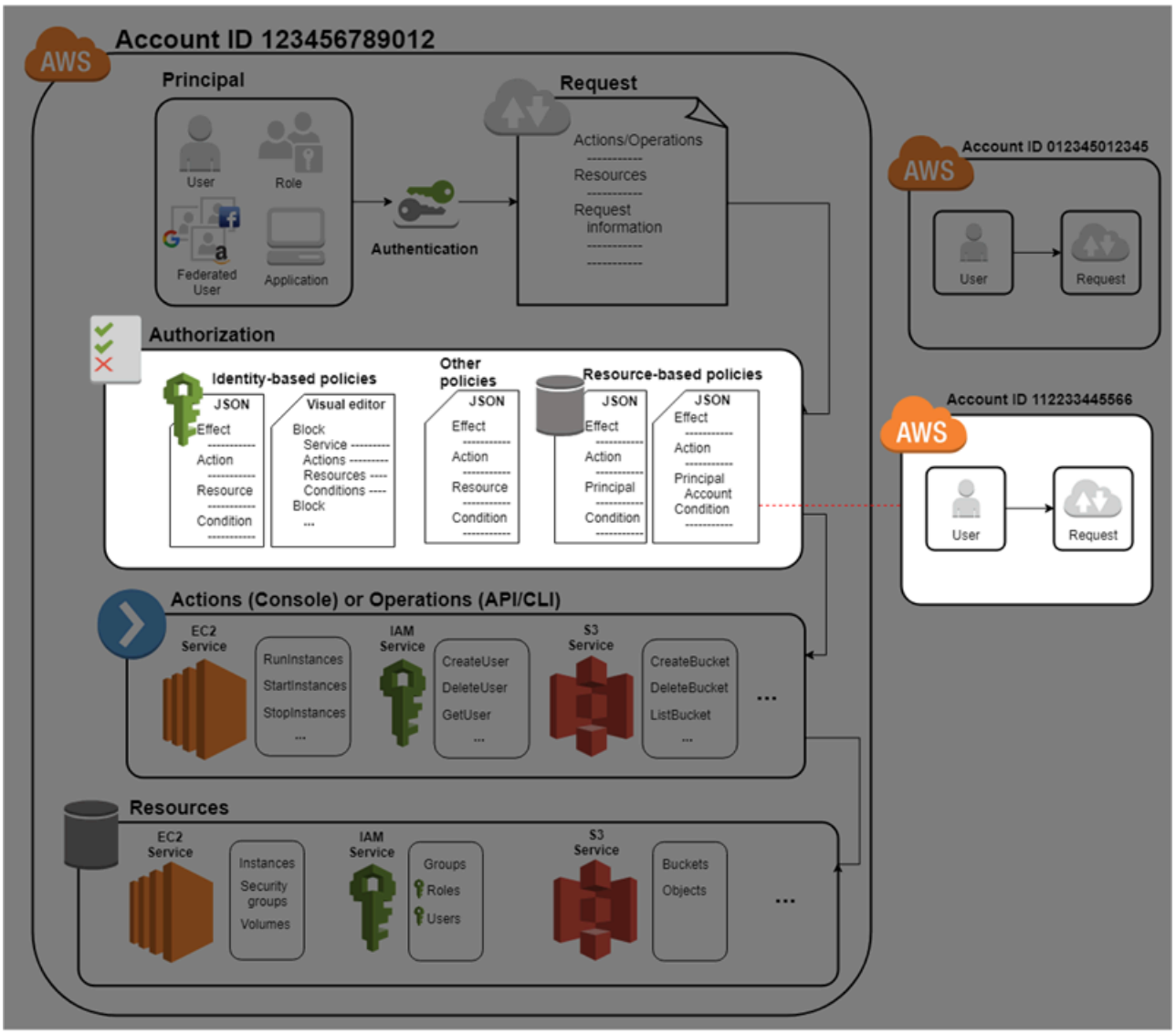
Am 21. September 2022 AWS wurden Änderungen am Verhalten der Vertrauensrichtlinie für Rollen vorgenommen, sodass in einer Richtlinie zur Vertrauensstellung für Rollen ausdrückliche Genehmigungen erforderlich sind, wenn sich eine Rolle von selbst annimmt. IAM-Rollen in der Legacy-Zulassungsliste für Verhaltensmuster verfügen über ein additionalEventData Feld explicitTrustGrant für `AssumeRole` Ereignisse. Der Wert von `explicitTrustGrant` ist false, wenn eine Rolle auf der Legacy-Zulassungsliste davon ausgeht, dass sie das alte Verhalten verwendet. Wenn eine Rolle auf der Legacy-Zulassungsliste sich selbst annimmt, das Verhalten der Rollenvertrauensrichtlinie jedoch so aktualisiert wurde, dass die Rolle explizit sich selbst übernehmen kann, `explicitTrustGrant` ist der Wert von true.

Nur eine sehr geringe Anzahl von IAM-Rollen steht auf der Zulassungsliste für das veraltete Verhalten, und dieses Feld ist in den CloudTrail Protokollen für diese Rollen nur vorhanden, wenn sie sich selbst übernehmen. In den meisten Fällen ist es nicht erforderlich, dass sich eine IAM-Rolle von selbst übernimmt. AWS empfiehlt, Ihre Prozesse, Ihren Code oder Ihre Konfigurationen zu aktualisieren, um dieses Verhalten zu beseitigen, oder Ihre Richtlinien für die Vertrauensstellung von Rollen zu aktualisieren, um dieses Verhalten ausdrücklich zuzulassen. Weitere Informationen finden Sie unter Ankündigung einer Aktualisierung des Verhaltens der IAM-Richtlinien zur Vertrauensstellung in Rollen.

Zugriffsmanagement für AWS Ressourcen

AWS Identity and Access Management (IAM) ist ein Webdienst, mit dem Sie den Zugriff auf Ressourcen sicher kontrollieren können. AWS Wenn ein Principal eine Anfrage stellt AWS, überprüft der AWS Durchsetzungscode, ob der Principal authentifiziert (angemeldet) und autorisiert ist (über Berechtigungen verfügt). Sie verwalten den Zugriff, AWS indem Sie Richtlinien erstellen und diese an IAM-Identitäten oder -Ressourcen anhängen. AWS Richtlinien sind JSON-Dokumente, in AWS denen, wenn sie an eine Identität oder Ressource angehängt werden, deren Berechtigungen definieren. Weitere Informationen zu diesen Richtlinienarten und ihrer Verwendung finden Sie unter Berechtigungen und Richtlinien in IAM.

Detaillierte Informationen zum Rest des Authentifizierungs- und Autorisierungsprozesses finden Sie unter Funktionsweise von IAM.



Während der Autorisierung überprüft der AWS Erzwingungscode anhand von Werten aus dem Anforderungskontext, ob die Richtlinien übereinstimmen, und bestimmt, ob die Anforderung zugelassen oder abgelehnt werden soll.

AWS überprüft jede Richtlinie, die für den Kontext der Anfrage gilt. Wenn eine einzelne Richtlinie die Anfrage ablehnt, AWS lehnt sie die gesamte Anfrage ab und beendet die Auswertung der Richtlinien. Dieser Vorgang wird als explizite Zugriffsverweigerung bezeichnet. Da Anforderungen standardmäßig verweigert werden, autorisiert IAM Ihre Anforderung nur dann, wenn jeder Teil Ihrer Anforderung von den anwendbaren Richtlinien erlaubt wird. Die Auswertungslogik für eine Anforderung in einem einzelnen Konto folgt diesen Regeln:

- Standardmäßig werden alle Anforderungen implizit verweigert. (Alternativ hat Root-Benutzer des AWS-Kontos standardmäßig vollen Zugriff.)

- Eine explizite Zugriffserlaubnis in einer identitätsbasierten oder ressourcenbasierten Richtlinie hat Vorrang vor diesem Standardwert.
- Wenn eine Berechtigungsgrenze, Organizations-SCP oder Sitzungsrichtlinie vorhanden ist, kann sie die Zugriffserlaubnis mit einer impliziten Zugriffsverweigerung überschreiben.
- Eine explizite Zugriffsverweigerung überschreibt jede Zugriffserlaubnis in einer Richtlinie.

AWS Genehmigt die Anfrage, nachdem Ihre Anfrage authentifiziert und autorisiert wurde. Wenn Sie eine Anforderung in einem anderen Konto initiieren müssen, muss eine Richtlinie in dem anderen Konto Ihnen den Zugriff auf die Ressource erlauben. Darüber hinaus muss die IAM-Entität, die Sie zum Erstellen der Anforderung verwenden, eine identitätsbasierte Richtlinie aufweisen, die die Anforderung zulässt.

Zugriffsmanagementressourcen

Weitere Informationen über Berechtigungen und zum Erstellen von Richtlinien finden Sie in folgenden Ressourcen:

Die folgenden Einträge im AWS Sicherheits-Blog behandeln gängige Methoden zum Schreiben von Richtlinien für den Zugriff auf Amazon S3 S3-Buckets und -Objekte.

- Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket
- Writing IAM policies: Zugriff auf benutzerspezifische Ordner in einem Amazon S3 Bucket gewähren
- IAM Policies and Bucket Policies and ACLs! Oh je! (Steuern des Zugriffs auf S3-Ressourcen)
- Eine Einführung in die Berechtigungen auf RDS-Ressourcenebene
- Entmystifizierung von EC2-Berechtigungen auf Ressourcenebene

Berechtigungen und Richtlinien in IAM

Sie verwalten den Zugriff in, AWS indem Sie Richtlinien erstellen und diese an IAM-Identitäten (Benutzer, Benutzergruppen oder Rollen) oder Ressourcen anhängen. AWS 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 IAM-Prinzipal (Benutzer oder Rolle) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. AWS unterstützt sechs Arten von Richtlinien: identitätsbasierte Richtlinien, ressourcenbasierte Richtlinien, Berechtigungsgrenzen, Organisations-SCPs, ACLs und Sitzungsrichtlinien.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Wenn eine Richtlinie die GetUserAktion beispielsweise zulässt, kann ein Benutzer mit dieser Richtlinie Benutzerinformationen von der AWS Management Console, oder der API abrufen. AWS CLI AWS Wenn Sie einen IAM-Benutzer erstellen, können Sie auswählen, ob Konsolen- oder Programmzugriff erlaubt sind. Wenn Konsolenzugriff erlaubt ist, kann sich der IAM-Benutzer mit seinen Anmeldeinformationen an der Konsole anmelden. Wenn der programmatische Zugriff zulässig ist, kann der Benutzer Zugriffsschlüssel verwenden, um mit der CLI oder der API zu arbeiten.

Richtlinientypen

Die folgenden Richtlinientypen, aufgelistet in der Reihenfolge von am häufigsten verwendeten bis zu seltener verwendeten, stehen für die Verwendung in AWS. Weitere Informationen finden Sie in den folgenden Abschnitten zu den einzelnen Richtlinientypen.

- Identitätsbasierte Richtlinien – Fügen Sie verwaltete und Inline-Richtlinien an IAM-Identitäten (Benutzer, Gruppen, zu denen Benutzer gehören, oder Rollen) an. Identitätsbasierte Richtlinien erteilen Berechtigungen für eine Identität.
- Ressourcenbasierte Richtlinien – Fügen Inline-Richtlinien an Ressourcen an. Die häufigsten Beispiele für ressourcenbasierte Richtlinien sind Amazon S3-Bucket-Richtlinien und Vertrauensrichtlinien für IAM-Rollen. Ressourcenbasierte Richtlinien erteilen dem Auftraggeber Berechtigungen, der in der Richtlinie angegeben ist. Auftraggeber können sich in demselben Konto wie die Ressource oder in anderen Konten befinden.
- Berechtigungsgrenzen – Verwenden Sie eine verwaltete Richtlinie als Berechtigungsgrenze für eine IAM-Entität (Benutzer oder Rolle). Diese Richtlinie definiert die maximalen Berechtigungen, die die identitätsbasierten Richtlinien einer Entität erteilen können, sie erteilt jedoch keine Berechtigungen. Berechtigungsgrenzen definieren nicht die maximalen Berechtigungen, die eine ressourcenbasierte Richtlinie einer Entität erteilen kann.
- SCPs für Organizations — Verwenden Sie eine AWS Organizations Service Control Policy (SCP), um die maximalen Berechtigungen für Kontomitglieder einer Organisation oder Organisationseinheit (OU) zu definieren. SCPs schränken Berechtigungen ein, die identitätsbasierte Richtlinien oder ressourcenbasierte Richtlinien Entitäten (Benutzern oder Rollen) innerhalb des Kontos erteilen, aber sie gewähren keine Berechtigungen.
- Zugriffssteuerungslisten (ACLs) – Verwenden Sie ACLs, um zu kontrollieren, welche Auftraggeber in anderen Konten auf eine Ressource zugreifen dürfen, an die die ACL angefügt ist. ACLs sind ähnlich wie ressourcenbasierten Richtlinien, obwohl sie der einzige Richtlinientyp sind, der die JSON-Richtliniendokumentstruktur nicht verwendet. ACLs sind kontoübergreifende

Berechtigungsrichtlinien, mit denen dem angegebenen Auftraggeber Berechtigungen erteilt werden. ACLs können keine Berechtigungen für Entitäten innerhalb desselben Kontos erteilen.

- Sitzungsrichtlinien — Übergeben Sie erweiterte Sitzungsrichtlinien, wenn Sie die AWS API AWS CLI oder verwenden, um eine Rolle oder einen Verbundbenutzer anzunehmen. Sitzungsrichtlinien begrenzen die Berechtigungen, die die identitätsbasierten Richtlinien der Rolle oder des Benutzers der Sitzung zu gewähren. Sitzungsrichtlinien begrenzen Berechtigungen für eine erstellte Sitzung, aber sie gewähren keine Berechtigungen. Weitere Informationen finden Sie unter Sitzungsrichtlinien

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die steuern, welche Aktionen eine Identität (Benutzer, Benutzergruppen und Rollen) für welche Ressourcen und unter welchen Bedingungen ausführen kann. Identitätsbasierte Richtlinien können weiter kategorisiert werden:

- Verwaltete Richtlinien — Eigenständige, identitätsbasierte Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können. AWS-Konto Es gibt zwei Typen von verwalteten Richtlinien:
  - AWS verwaltete Richtlinien — Verwaltete Richtlinien, die von erstellt und verwaltet werden. AWS
  - Vom Kunden verwaltete Richtlinien – Dies sind verwaltete Richtlinien, die Sie in Ihrem AWS-Konto erstellen und verwalten. Vom Kunden verwaltete Richtlinien bieten eine genauere Kontrolle über Ihre Richtlinien als AWS verwaltete Richtlinien.
- Eingebundene Richtlinien – Richtlinien, die Sie direkt zu einem einzelnen Benutzer, einer einzelnen Gruppe oder einer einzelnen Rolle hinzufügen. Inline-Richtlinien sorgen für eine strikte one-to-one Beziehung zwischen einer Richtlinie und einer Identität. Sie werden gelöscht, wenn Sie die Identität löschen.

Informationen darüber, wie Sie entscheiden können, ob Sie eine verwaltete Richtlinie oder eine Inline-Richtlinie verwenden sollten, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource wie einen Amazon S3-Bucket anfügen. Diese Richtlinien erteilen dem angegebenen Auftraggeber

die Berechtigung zum Ausführen bestimmter Aktionen für diese Ressource und definiert, unter welchen Bedingungen dies gilt. Ressourcenbasierte Richtlinien sind Inline-Richtlinien. Es gibt keine verwalteten ressourcenbasierten Richtlinien.

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Auftraggeber in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn der Prinzipal und die Ressource getrennt sind AWS-Konten, müssen Sie außerdem eine identitätsbasierte Richtlinie verwenden, um dem Prinzipal Zugriff auf die Ressource zu gewähren. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Auftraggeber in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Eine Schritt-für-Schritt-Anleitung für die Gewährung von dienstübergreifendem Zugriff finden Sie unter Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen.

Der IAM-Service unterstützt nur eine Art von ressourcenbasierter Richtlinie, die als Rollen-Vertrauensrichtlinie bezeichnet wird, die einer IAM-Rolle zugewiesen ist. Eine IAM-Rolle ist sowohl eine Identität als auch eine Ressource, die ressourcenbasierte Richtlinien unterstützt. Aus diesem Grund müssen Sie einer IAM-Rolle eine Vertrauensrichtlinie und eine identitätsbasierte Richtlinie anfügen. Vertrauensrichtlinien definieren, welche Auftraggeber-Entitäten (Konten, Benutzer, Rollen und Verbundbenutzer) die Rolle übernehmen können. Informationen darüber, inwieweit sich IAM-Rollen von anderen ressourcenbasierten Richtlinien unterscheiden, finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.

Informationen darüber, welche anderen Services ressourcenbasierte Richtlinien unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Weitere Informationen zu ressourcenbasierten Richtlinien finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien. Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

IAM-IBerechtigungsgrenzen

Eine Berechtigungsgrenze ist ein erweitertes Feature, die Ihnen ermöglicht, die maximalen Berechtigungen festzulegen, die eine identitätsbasierte Richtlinie einer IAM-Entität gewähren kann. Wenn Sie eine Berechtigungsgrenze für eine Entität festlegen, kann die Entität nur die Aktionen durchführen, die sowohl von den identitätsbasierten Richtlinien als auch den Berechtigungsgrenzen erlaubt werden. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle als Auftraggeber 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.

Service-Kontrollrichtlinien (SCPs)

AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung der Ressourcen 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. SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) festlegen. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft.

Weitere Informationen über Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations -Leitfaden.

Zugriffssteuerungslisten (ACLs)

Zugriffskontrolllisten (ACLs) sind Service-Richtlinien, mit denen Sie steuern können, welche Auftraggeber in einem anderen Konto auf eine Ressource zugreifen können. ACLs können nicht verwendet werden, um den Zugriff für einen Auftraggeber innerhalb desselben Kontos zu steuern. ACLs sind ähnlich wie ressourcenbasierten Richtlinien, obwohl sie der einzige Richtlinientyp sind, der das JSON-Richtliniendokumentformat nicht verwendet. 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 Developer Guide.

Sitzungsrichtlinien

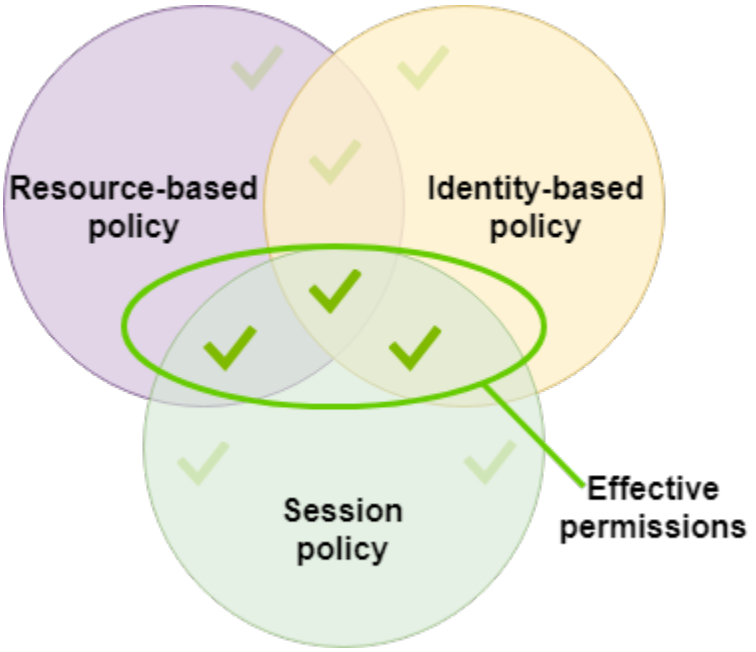
Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie programmgesteuert eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Die Berechtigungen für eine Sitzung stammen aus den identitätsbasierten Richtlinien für die IAM-Entität (Benutzer oder Rolle), die zum Erstellen der Sitzung und der Sitzungsrichtlinie verwendet wurde. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft.

Sie können eine Rollensitzung erstellen und eine Sitzungsrichtlinie programmgesteuert mithilfe der API-Operationen `AssumeRole`, `AssumeRoleWithSAML` oder `AssumeRoleWithWebIdentity` übergeben. Sie können ein einzelnes JSON-Inline-Sitzungsrichtliniendokument mit dem `Policy`-Parameter übergeben. Sie können mit dem Parameter `PolicyArns` bis zu 10 verwaltete

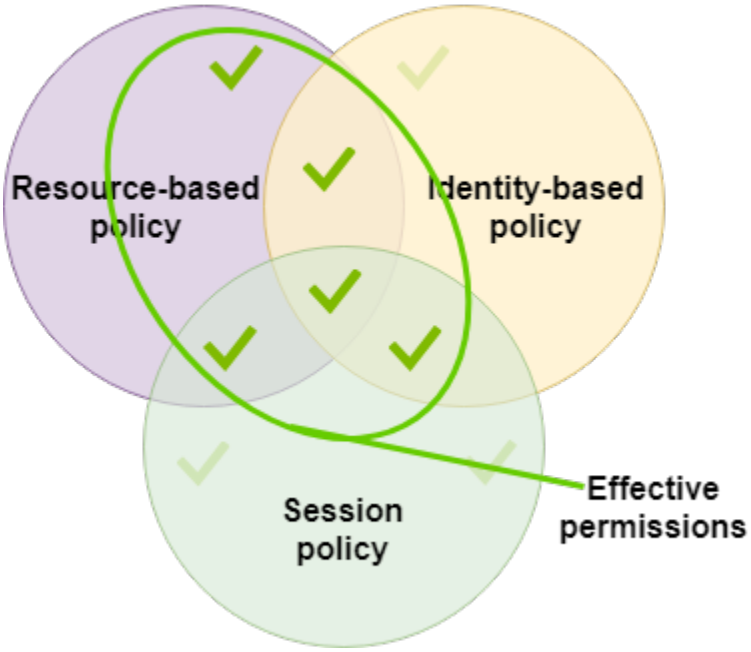
Sitzungsrichtlinien angeben. Weitere Informationen zum Erstellen einer Rollensitzung finden Sie unter Anfordern temporärer Sicherheitsanmeldeinformationen.

Wenn Sie eine Sitzung für Verbundbenutzer erstellen, verwenden Sie einen Zugriffsschlüssel des IAM-Benutzers, um die `GetFederationToken`-API-Operation programmgesteuert aufzurufen. Außerdem müssen Sie Sitzungsrichtlinien übergeben. Die resultierenden Sitzungsberechtigungen stammen aus der identitätsbasierten Richtlinie und der Sitzungsrichtlinie. Weitere Informationen zum Erstellen von einer Sitzung für Verbundbenutzer finden Sie unter GetFederationToken - Verbund durch einen benutzerdefinierten Identity Broker.

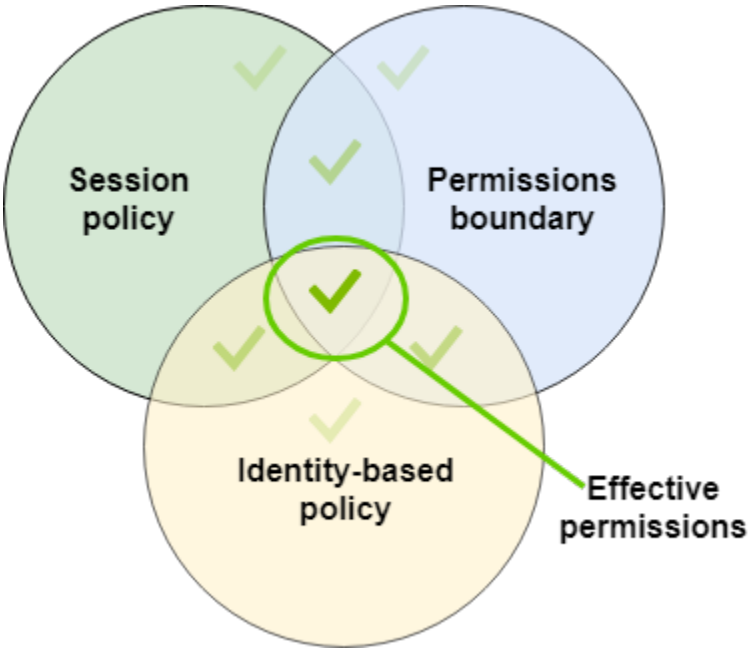
Eine ressourcenbasierte Richtlinie kann den ARN des Benutzers oder der Rolle als Auftraggeber angeben. In diesem Fall werden die Berechtigungen aus der ressourcenbasierten Richtlinie an die Rolle oder die identitätsbasierte Richtlinie des Benutzers vor der Erstellung der Sitzung hinzugefügt. Die Sitzungsrichtlinie beschränkt die Gesamtzahl der Berechtigungen, die von der ressourcenbasierten Richtlinie und der identitätsbasierten Richtlinie erteilt werden. Die resultierenden Sitzungsberechtigungen ergeben sich aus den Sitzungsrichtlinien und entweder der ressourcenbasierten Richtlinie oder der identitätsbasierten Richtlinie.



Eine ressourcenbasierte Richtlinie kann den ARN der Sitzung als Auftraggeber angeben. In diesem Fall werden die Berechtigungen aus der ressourcenbasierten Richtlinie nach dem Erstellen der Sitzung hinzugefügt. Die Berechtigungen der ressourcenbasierte Richtlinie werden nicht von der Sitzungsrichtlinie eingeschränkt. Die resultierende Sitzung verfügt über alle Berechtigungen der ressourcenbasierten Richtlinie plus die Berechtigungen, die sowohl von der identitätsbasierten Richtlinie als auch von der Sitzungsrichtlinie erteilt werden.



Ein Berechtigungsgrenze kann für einen Benutzer oder eine Rolle die Höchstzahl der Berechtigungen festlegen, die für die Erstellung der Sitzung verwendet werden. In diesem Fall ergeben sich die resultierenden Sitzungsberechtigungen aus der Sitzungsrichtlinie, der Berechtigungsgrenze und der identitätsbasierten Richtlinie. Eine Berechtigungsgrenze schränkt jedoch nicht die Berechtigungen ein, die von einer ressourcenbasierten Richtlinie gewährt werden, die den ARN der resultierenden Sitzung angibt.



Richtlinien und Stammbenutzer

Die Root-Benutzer des AWS-Kontos wird von einigen Richtlinientypen beeinflusst, von anderen jedoch nicht. Sie können dem Stammbenutzer keine identitätsbasierten Richtlinien zuweisen und Sie können die Berechtigungsgrenze für den Stammbenutzer nicht festlegen. Sie können jedoch die Stammbenutzer; als Auftraggeber in einer ressourcenbasierten Richtlinie oder einer ACL angeben. Ein Stammbenutzer ist immer noch das Mitglied eines Kontos. Wenn dieses Konto Mitglied einer Organisation in ist AWS Organizations, ist der Root-Benutzer von allen SCPs für das Konto betroffen.

Übersicht über JSON-Richtlinien

Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Identitätsbasierte Richtlinien und Richtlinien zum Festlegen von Berechtigungsgrenzen sind JSON-Richtliniendokumente, die Sie einem Benutzer oder einer Rolle anfügen. Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. SCPs sind JSON-Richtliniendokumente mit eingeschränkter Syntax, die Sie einer AWS Organizations Organisationseinheit (OU) zuordnen. ACLs werden auch einer Ressource angefügt. Sie müssen jedoch einen anderen Syntax verwenden. Sitzungsrichtlinien sind JSON-Richtlinien, die Sie bereitstellen, wenn Sie in einer Sitzung eine Rolle oder einen Verbundbenutzer übernehmen.

Es ist nicht notwendig, dass Sie die JSON-Syntax verstehen. Sie können den visuellen Editor in verwenden, AWS Management Console um vom Kunden verwaltete Richtlinien zu erstellen und zu bearbeiten, ohne jemals JSON verwenden zu müssen. Wenn Sie jedoch eingebundene Richtlinien für Gruppen oder komplexe Richtlinien verwenden, müssen Sie diese Richtlinien im JSON-Editor unter Verwendung der Konsole erstellen und bearbeiten. Weitere Informationen zur Verwendung des visuellen Editors finden Sie unter Erstellen von IAM-Richtlinien und Bearbeiten von IAM-Richtlinien.

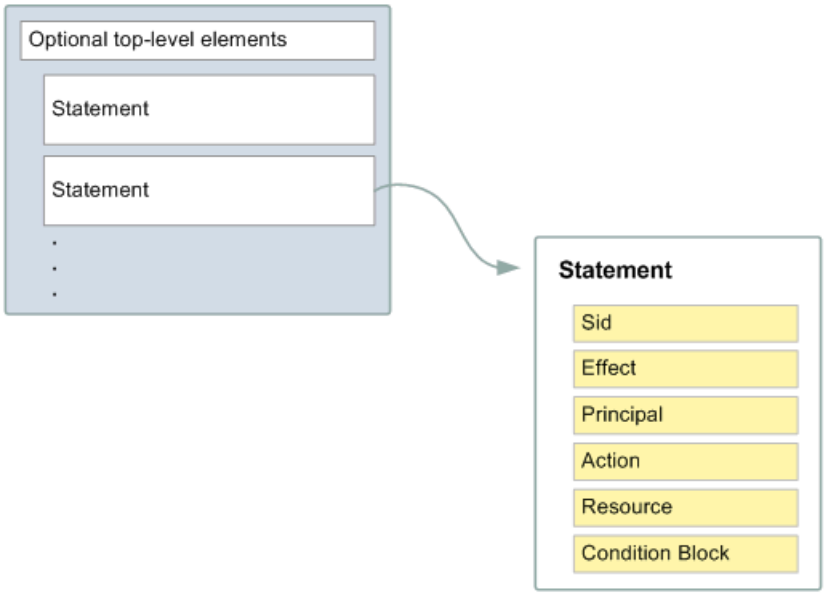
Wenn Sie eine JSON-Richtlinie erstellen oder bearbeiten, kann IAM eine Richtlinienvalidierung durchführen, um Ihnen beim Erstellen einer effektiven Richtlinie zu helfen. IAM identifiziert JSON-Syntaxfehler, während IAM Access Analyzer zusätzliche Richtlinienüberprüfungen mit Empfehlungen zur weiteren Verfeinerung Ihrer Richtlinien bietet. Weitere Informationen zur Richtlinienvalidierung finden Sie unter Validieren von IAM-Richtlinien. Weitere Informationen zu IAM Access Analyzer Richtlinienvalidierungen und umsetzbaren Empfehlungen finden Sie unter IAM Access Analyzer-Richtlinienvalidierung.

JSON-Richtliniendokumentstruktur

Wie in der folgenden Abbildung dargestellt, umfasst ein JSON-Richtliniendokument die folgenden Elemente:

- Optionale richtlinienweite Informationen oben im Dokument.
- Eine oder mehrere einzelne Anweisungen

Jede Anweisung enthält Angaben über die jeweilige Berechtigung. Wenn eine Richtlinie mehrere Aussagen enthält, AWS wendet sie bei deren Auswertung `OR` eine logische Anweisung für alle Anweisungen an. Wenn mehrere Richtlinien auf eine Anfrage AWS zutreffen, wird bei der Bewertung eine logische Regel `OR` auf alle diese Richtlinien angewendet.



Die Informationen in einer Anweisung sind in einer Reihe von Elementen enthalten.

- Version – Geben Sie die Version der Richtliniensprache an, die Sie verwenden möchten. Wir empfehlen, dass Sie die neueste Version `2012-10-17` verwenden. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Version.
- Statement – Verwenden Sie dieses Haupt-Richtlinienelement als Container für die folgenden Elemente. Sie können mehrere Anweisungen in eine Richtlinie aufnehmen.
- Sid (Optional) – Fügen Sie eine optionale Statement-ID an, um Ihre Anweisungen unterscheiden zu können.
- Effect – Verwenden Sie `Allow` oder `Deny`, um anzugeben, ob die Richtlinie den Zugriff erlaubt oder verweigert.
- Principal (Nur in einigen Fällen erforderlich) – Wenn Sie eine ressourcenbasierte Richtlinie erstellen, müssen Sie das Konto, den Benutzer, die Rolle oder den Verbundbenutzer angeben, für das Sie den Zugriff zulassen oder verweigern möchten. Wenn Sie eine IAM-Berechtigung zum

Anfügen für einen Benutzer oder eine Rolle erstellen, können Sie dieses Element nicht aufnehmen. Der Auftraggeber wird als dieser Benutzer oder diese Rolle impliziert.

- Action – Binden Sie eine Liste der Aktionen ein, die durch die Richtlinie erlaubt oder verweigert werden.
- Resource (Nur in einigen Fällen erforderlich) – Wenn Sie eine IAM-Berechtigungsrichtlinie erstellen, müssen Sie eine Liste der Ressourcen angeben, für die die Aktionen gelten. Wenn Sie eine ressourcenbasierte Richtlinie erstellen, ist dieses Element optional. Wenn Sie dieses Element nicht einschließen, ist die Ressource, auf die die Aktion angewendet wird, die Ressource, der die Richtlinie angefügt ist.
- Condition (Optional) – Geben Sie die Bedingungen an, unter denen die Richtlinie die Berechtigung erteilt.

Weitere Informationen zu diesen und erweiterten Richtlinienelementen finden Sie unter IAM-JSON-Richtlinienelementreferenz.

Mehrere Anweisungen und mehrere Richtlinien

Wenn Sie mehr als eine Berechtigung für eine Entity (Benutzer, Gruppe oder Rolle) definieren möchten, können Sie in einer einzigen Richtlinie mehrere Anweisungen verwenden. Sie können auch mehrere Richtlinien anfügen. Wenn Sie versuchen, mehrere Berechtigungen in einer einzigen Anweisung zu definieren, gewährt Ihre Richtlinie möglicherweise nicht den Zugriff, den Sie erwarten. Wir empfehlen, dass Sie die Richtlinien nach Ressourcentypen aufteilen.

Aufgrund der eingeschränkten Größe der Richtlinienmüssen Sie für komplexere Berechtigungen möglicherweise mehrere Richtlinien verwenden. Es ist auch sinnvoll, funktionale Gruppierungen von Berechtigungen in einer separaten, vom Kunden verwalteten Richtlinie zu erstellen. Beispiel: Erstellen Sie eine Richtlinie für die IAM-Benutzerverwaltung, eine für die Selbstverwaltung und eine weitere Richtlinie für die Verwaltung von S3-Buckets. Unabhängig von der Kombination mehrerer Aussagen und mehrerer Richtlinien werden Ihre Richtlinien auf die gleiche Weise AWS bewertet.

Die folgende Richtlinie enthält beispielsweise drei Anweisungen, von denen jede einen eigenen Satz Berechtigungen innerhalb eines einzelnen Kontos definiert. Die Anweisungen definieren Folgendes:

- Mit der ersten Anweisung mit der `Sid` (Statement-ID) `FirstStatement` kann der Benutzer mit der zugeordneten Richtlinie das eigene Passwort ändern. Das `Resource`-Element in dieser Anweisung ist * (das bedeutet "alle Ressourcen"). In der Praxis wirkt sich die API-Operation `ChangePassword` (oder der entsprechende CLI-Befehl `change-password`) praktisch nur auf das Passwort des Benutzers aus, der die Anforderung gestellt hat.

- Über die zweite Anweisung kann der Benutzer alle Amazon S3-Buckets in seinem AWS-Konto auflisten. Das `Resource`-Element in dieser Anweisung ist `"*"` (das bedeutet "alle Ressourcen"). Da aber Richtlinien den Zugriff nicht auf Ressourcen in anderen Konten erteilen können, kann der Benutzer nur die Buckets in seinem eigenen AWS-Konto auflisten.
- Durch die dritte Anweisung kann der Benutzer beliebige Objekte auflisten und abrufen, die sich in einem Bucket mit dem Namen `confidential-data` befinden. Allerdings ist dazu die Authentifizierung des Benutzers durch eine Multi-Factor Authentication (MFA) erforderlich. Das `Condition`-Element in der Richtlinie erzwingt die MFA-Authentifizierung.

  Wenn eine Richtlinienanweisung ein `Condition`-Element enthält, ist die Anweisung nur dann wirksam, wenn das `Condition`-Element mit „true“ ausgewertet wird. In diesem Fall wird `Condition` mit „true“ ausgewertet, wenn der Benutzer durch MFA authentifiziert ist. Wenn der Benutzer nicht durch MFA authentifiziert ist, wird `Condition` mit „false“ ausgewertet. In diesem Fall wird die dritte Anweisung in dieser Richtlinie nicht wirksam, sodass der Benutzer keinen Zugriff auf den `confidential-data`-Bucket hat.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::confidential-data",
```

```
        "arn:aws:s3:::confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Beispiele für JSON-Richtliniensyntax

Die folgenden identitätsbasierte Richtlinie erlaubt dem implizierten Auftraggeber, einen einzelnen Amazon S3-Bucket mit dem Namen `example_bucket` aufzulisten:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Die folgende ressourcenbasierte Richtlinie kann einem Amazon S3-Bucket angefügt werden. Die Richtlinie ermöglicht es Mitgliedern einer bestimmten Person AWS-Konto , alle Amazon S3 S3-Aktionen in dem genannten Bucket durchzuführen`mybucket`. Sie erlaubt alle Aktionen, die für einen Bucket oder die darin enthaltenen Objekte durchgeführt werden können. (Da die Richtlinie nur dem Konto eine Vertrauensstellung einräumt, müssen den einzelnen Benutzer nach wie vor Berechtigungen für die angegebenen Amazon S3-Aktionen gewährt werden.)

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```

Beispielrichtlinien für gebräuchliche Szenarien finden Sie unter Beispiele für identitätsbasierte Richtlinien in IAM.

Gewähren von geringsten Rechten

Wenn Sie IAM-Richtlinien erstellen, befolgen Sie die Standard-Sicherheitsmaßnahmen, die für das Erteilen von geringsten Rechten gelten, d. h. es werden nur die Berechtigungen erteilt, die zum Durchführen einer Aufgabe erforderlich sind. Bestimmen Sie, was Benutzer und Rollen tun müssen, und gestalten Sie dann entsprechende Richtlinien, mit denen die Benutzer nur diese Aufgaben ausführen können.

Beginnen Sie mit einem Mindestsatz von Berechtigungen und gewähren Sie zusätzliche Berechtigungen wie erforderlich. Dies ist sicherer, als mit Berechtigungen zu beginnen, die zu weit gefasst sind, und dann später zu versuchen, sie zu begrenzen.

Als Alternative zu Least Privilege können Sie verwaltete AWS -Richtlinien oder Richtlinien mit Wildcard-*-Berechtigungen verwenden, um mit Richtlinien zu beginnen. Berücksichtigen Sie das Sicherheitsrisiko, wenn Sie Ihren Auftraggeber mehr Berechtigungen gewähren, als sie für ihre Arbeit benötigen. Überwachen Sie diese Auftraggeber, um zu erfahren, welche Berechtigungen sie verwenden. Schreiben Sie dann Richtlinien mit den geringsten Berechtigungen.

IAM bietet verschiedene Optionen, mit denen Sie die von Ihnen erteilten Berechtigungen verfeinern können.

- Zugriffsebenengruppierungen – Mit Zugriffsebenengruppierungen können Sie die von einer Richtlinie gewährte Zugriffsebene nachvollziehen. Richtlinienaktionen werden als `List`, `Read`, `Write`, `Permissions management`, oder `Tagging` klassifiziert. Beispielsweise können Sie Aktionen aus den Zugriffsebenen `List` und `Read` wählen, um Ihren Benutzern schreibgeschützten Zugriff zu erteilen. Informationen zur Verwendung von Richtlinienzusammenfassungen und Erläuterungen der Berechtigungen auf Zugriffsebene finden Sie unter Zugriffsebenen in Richtlinienzusammenfassungen verstehen.
- Validieren Ihrer Richtlinien – Sie können die Richtlinienvalidierung mit IAM Access Analyzer durchführen, wenn Sie JSON-Richtlinien erstellen und bearbeiten. Wir empfehlen, alle vorhandenen Richtlinien zu überprüfen und zu validieren. IAM Access Analyzer bietet über 100 Richtlinienüberprüfungen zur Validierung Ihrer Richtlinien. Es generiert Sicherheitswarnungen, wenn eine Anweisung in Ihrer Richtlinie den Zugriff zulässt, den wir als übermäßig freizügig betrachten. Sie können die umsetzbaren Empfehlungen verwenden, die durch die Sicherheitswarnungen bereitgestellt werden, wenn Sie daran arbeiten, die geringsten

Berechtigungen zu erteilen. Weitere Informationen zu den von IAM Access Analyzer bereitgestellten Richtlinienprüfungen finden Sie unter IAM Access Analyzer-Richtlinienvalidierung.

- Generieren einer Richtlinie basierend auf -Zugriffsaktivitäten – Um die Berechtigungen zu verfeinern, die Sie gewähren, können Sie eine IAM-Richtlinie generieren, die auf der Zugriffsaktivität für eine IAM-Entität (Benutzer oder Rolle) basiert. IAM Access Analyzer überprüft Ihre AWS CloudTrail Protokolle und generiert eine Richtlinienvorlage, die die Berechtigungen enthält, die von der Entität in dem von Ihnen angegebenen Zeitraum verwendet wurden. Sie können die Vorlage verwenden, um eine verwaltete Richtlinie mit definierten Berechtigungen zu erstellen und sie dann an die IAM-Rolle anzuhängen. Auf diese Weise gewähren Sie nur die Berechtigungen, die der Benutzer oder die Rolle benötigt, um mit AWS Ressourcen für Ihren speziellen Anwendungsfall zu interagieren. Weitere Informationen hierzu finden Sie unter Generieren von Richtlinien basierend auf Zugriffsaktivitäten.
- Verwenden der Informationen zum letzten Zugriff – Ein weiteres Feature, die bei der Wahrung der geringsten Berechtigung helfen kann, sind Informationen über den letzten Zugriff. Sie können diese Informationen auf der Registerkarte Access Advisor (Advisor aufrufen) auf der Detailseite für einen IAM-Benutzer, eine Gruppe, eine Rolle oder eine Richtlinie in der -Konsole anzeigen. Die Informationen über den letzten Zugriff enthalten Informationen über einige Aktionen, auf die zuletzt für Amazon EC2, IAM, Lambda und Amazon S3 zugegriffen wurde. Wenn Sie sich mit den Anmeldeinformationen für das AWS Organizations Verwaltungskonto anmelden, können Sie die Informationen zum letzten Zugriff auf den Dienst im AWS OrganizationsBereich der IAM-Konsole einsehen. Sie können die AWS API AWS CLI oder auch verwenden, um einen Bericht über die zuletzt abgerufenen Informationen für Entitäten oder Richtlinien in IAM oder Organizations abzurufen. Mit diesen Informationen können Sie unnötige Berechtigungen identifizieren, sodass Sie die IAM- oder Organizations-Richtlinien besser an die Regel der geringsten Rechte anpassen können. Weitere Informationen finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.
- Kontoereignisse überprüfen in AWS CloudTrail — Um die Zugriffsrechte weiter zu reduzieren, können Sie die Ereignisse Ihres Kontos im AWS CloudTrail Ereignisverlauf einsehen. CloudTrail Ereignisprotokolle enthalten detaillierte Ereignisinformationen, anhand derer Sie die Berechtigungen der Richtlinie einschränken können. Die Protokolle enthalten nur die Aktionen und Ressourcen, die Ihre IAM-Entitäten benötigen. Weitere Informationen finden Sie im AWS CloudTrail Benutzerhandbuch unter CloudTrail Ereignisse in der CloudTrail Konsole anzeigen.

Weitere Informationen finden Sie in den folgenden Richtlinienthemen für einzelne Services, die Beispiele für das Schreiben von Richtlinien für servicespezifische Ressourcen bieten.

- Amazon DynamoDB Authentifizierung und Zugriffskontrolle im Developer Guide für Amazon DynamoDB
- Verwendung von Bucket-Richtlinien und Benutzerrichtlinien im Benutzerhandbuch für Amazon Simple Storage Service
- Übersicht über Access Control List (ACL) im Benutzerhandbuch für Amazon Simple Storage Service

Verwaltete Richtlinien und eingebundene Richtlinien

Wenn Sie die Berechtigungen für eine Identität in IAM festlegen, müssen Sie entscheiden, ob Sie eine verwaltete AWS -Richtlinie, eine vom Kunden verwaltete Richtlinie oder eine Inline-Richtlinie verwenden. In den folgenden Themen erhalten Sie weitere Informationen zu den einzelnen Arten identitätsbasierter Richtlinien und deren Verwendung.

Themen

- AWS verwaltete Richtlinien
- Kundenverwaltete Richtlinien
- Eingebundene Richtlinien
- Auswahl zwischen verwalteten und eingebundenen Richtlinien
- Erste Schritte mit verwalteten Richtlinien
- So konvertieren Sie eine eingebundene Richtlinie in eine verwaltete Richtlinie
- Veraltete verwaltete Richtlinien AWS

AWS verwaltete Richtlinien

Bei einer von AWS verwalteten Richtlinie handelt es sich um eine eigenständige Richtlinie, die von AWS erstellt und verwaltet wird. Eigenständige Richtlinie bedeutet, dass die Richtlinie ihren eigenen Amazon-Ressourcennamen (ARN) hat, der den Richtliniennamen enthält. `arn:aws:iam::aws:policy/IAMReadOnlyAccess`Ist beispielsweise eine AWS verwaltete Richtlinie. Weitere Informationen zu ARNs finden Sie unter IAM-ARNs. Eine Liste der AWS verwalteten Richtlinien für AWS-Services finden Sie unter AWS Verwaltete Richtlinien.

AWS Mit verwalteten Richtlinien können Sie Benutzern, Gruppen und Rollen bequem die entsprechenden Berechtigungen zuweisen. Das ist schneller, als selbst die Richtlinien zu schreiben und beinhaltet Berechtigungen für viele häufige Anwendungsfälle.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. AWS aktualisiert gelegentlich die in einer AWS verwalteten Richtlinie definierten Berechtigungen. In diesem Fall AWS wirkt sich das Update auf alle Prinzipalentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS ist am wahrscheinlichsten, dass eine AWS verwaltete Richtlinie aktualisiert wird, wenn ein neuer AWS Dienst gestartet wird oder neue API-Aufrufe für bestehende Dienste verfügbar werden. Die so genannte AWS verwaltete Richtlinie ReadOnlyAccessbietet beispielsweise schreibgeschützten Zugriff auf alle AWS Dienste und Ressourcen. Wenn ein neuer Dienst AWS gestartet wird, wird die ReadOnlyAccessRichtlinie AWS aktualisiert, um schreibgeschützte Berechtigungen für den neuen Dienst hinzuzufügen. Die aktualisierten Berechtigungen werden auf alle Auftraggeber-Entitäten angewendet, an die die Richtlinie angefügt ist.

AWS Verwaltete Richtlinien mit vollem Zugriff definieren Berechtigungen für Dienstadministratoren, indem sie Vollzugriff auf einen Dienst gewähren.

- AmazonDynamoDB FullAccess
- ICH BIN FullAccess

Von Power-Usern AWS verwaltete Richtlinien bieten vollen Zugriff auf AWS Dienste und Ressourcen, ermöglichen jedoch nicht die Verwaltung von Benutzern und Gruppen.

- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

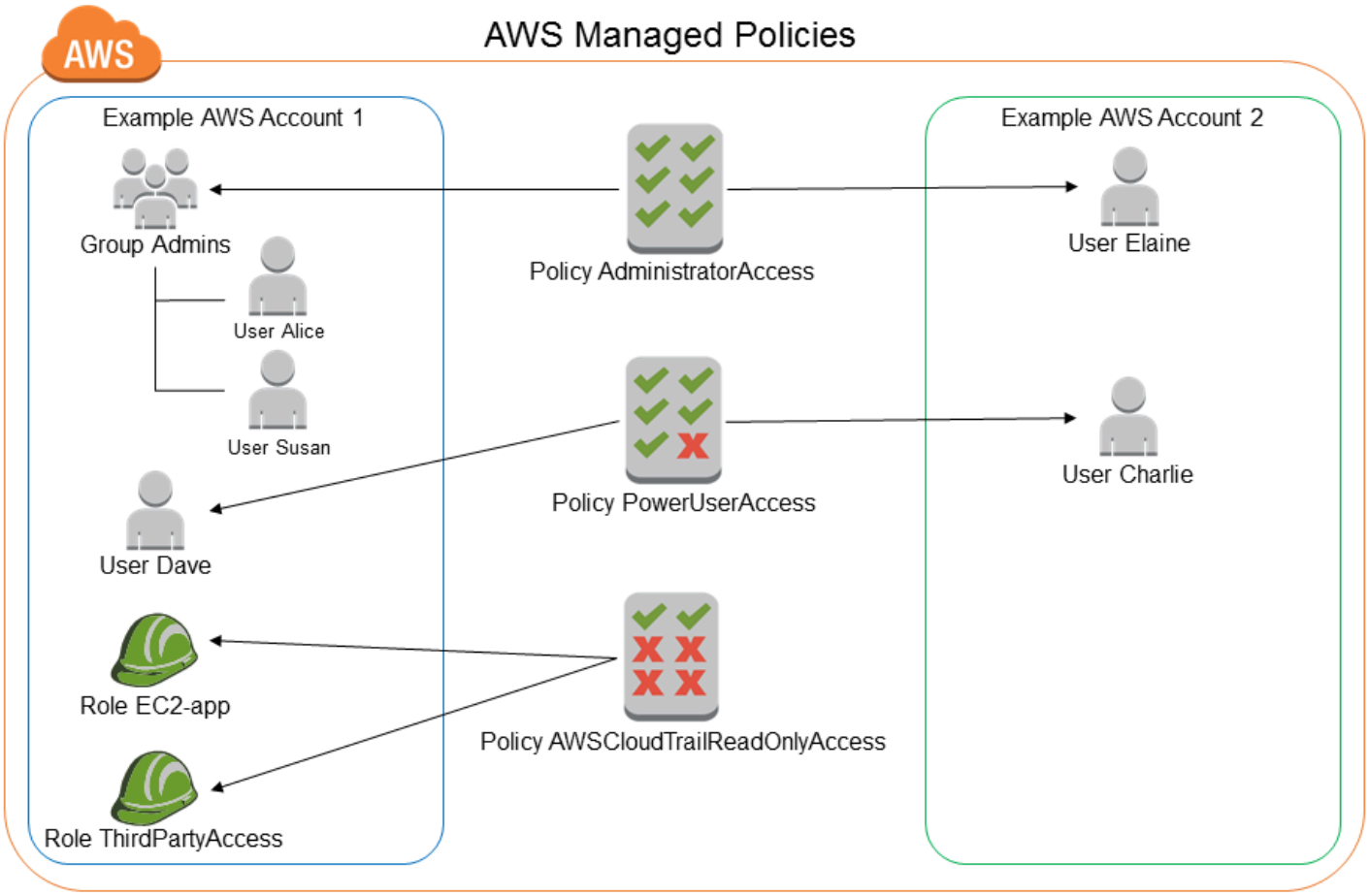
AWS Verwaltete Richtlinien mit teilweisem Zugriff bieten bestimmte Zugriffsebenen für AWS Dienste, ohne dass Berechtigungen für die Rechteverwaltung auf Zugriffsebene gewährt werden.

- AmazonMobileAnalyticsWriteOnlyAccess
- Amazon EC2 ReadOnlyAccess

Eine besonders nützliche Kategorie AWS verwalteter Richtlinien sind solche, die für berufliche Funktionen konzipiert sind. Diese Richtlinien sind genau an häufig genutzte Jobfunktionen in der IT-Branche angepasst und ermöglichen es, Berechtigungen für diese Funktionen zu gewähren. Ein entscheidender Vorteil der Verwendung von Richtlinien für berufliche Funktionen besteht darin, dass sie bei AWS der Einführung neuer Dienste und API-Operationen beibehalten und aktualisiert werden. Die AdministratorAccessJob-Funktion bietet beispielsweise vollen Zugriff und die Delegierung von

Berechtigungen für jeden Dienst und jede Ressource in AWS. Wir empfehlen, dass diese Richtlinie nur für den Kontoadministrator verwendet wird. Power-User, die vollen Zugriff auf alle Services mit Ausnahme des eingeschränkten Zugriffs auf IAM und Organizations benötigen, sollten die PowerUserAccessJob-Funktion verwenden. Eine Liste und Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie unter AWS verwaltete Richtlinien für Jobfunktionen.

Das folgende Diagramm veranschaulicht AWS verwaltete Richtlinien. Das Diagramm zeigt drei AWS verwaltete Richtlinien: AdministratorAccessPowerUserAccess, und AWS CloudTrailReadOnlyAccess. Beachten Sie, dass eine einzelne AWS verwaltete Richtlinie an verschiedene Haupteinheiten und an verschiedene AWS-Konten Haupteinheiten in einer einzigen Richtlinie angehängt werden kann AWS-Konto.



Kundenverwaltete Richtlinien

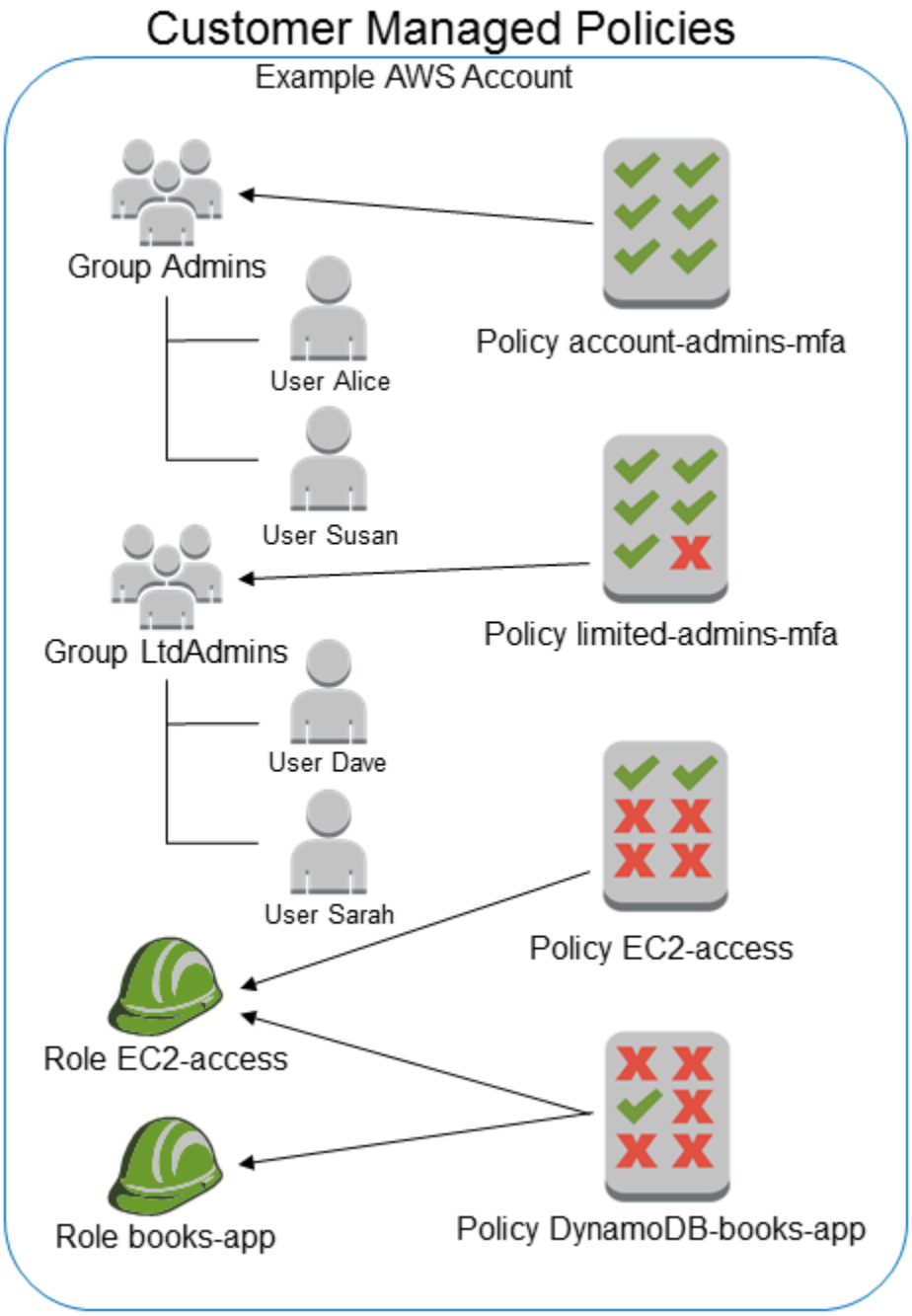
Sie können eigene eigenständige Richtlinien erstellen AWS-Konto , die Sie an Prinzipalentitäten (Benutzer, Gruppen und Rollen) anhängen können. Sie erstellen diese vom Kunden verwalteten Richtlinien für Ihre spezifischen Anwendungsfälle und können sie beliebig oft ändern und aktualisieren. Wie bei AWS verwalteten Richtlinien gewähren Sie, wenn Sie einer Prinzipalentität

eine Richtlinie zuordnen, der Entität die in der Richtlinie definierten Berechtigungen. Wenn Sie Berechtigungen in der Richtlinie aktualisieren, werden die Änderungen auf alle Prinzipal-Entitäten angewendet, an die die Richtlinie angefügt ist.

Eine hervorragende Methode zum Erstellen einer vom Kunden verwalteten Richtlinie besteht darin, zunächst eine vorhandene, von AWS verwaltete Richtlinie zu kopieren. So wissen Sie, dass die Richtlinie zu Beginn richtig ist und Sie sie nur an Ihre Umgebung anpassen müssen.

Die vom Kunden verwalteten Richtlinien sind in der folgenden Abbildung dargestellt. Bei der jeweiligen Richtlinie handelt es sich um eine Entität in IAM mit ihrem eigenen Amazon-Ressourcenname (ARN), der den Richtliniennamen enthält. Beachten Sie, dass ein und dieselbe Richtlinie mehreren Hauptentitäten zugeordnet werden kann. So ist beispielsweise dieselbe DynamoDB-books-app-Richtlinie zwei verschiedenen IAM-Rollen zugeordnet.

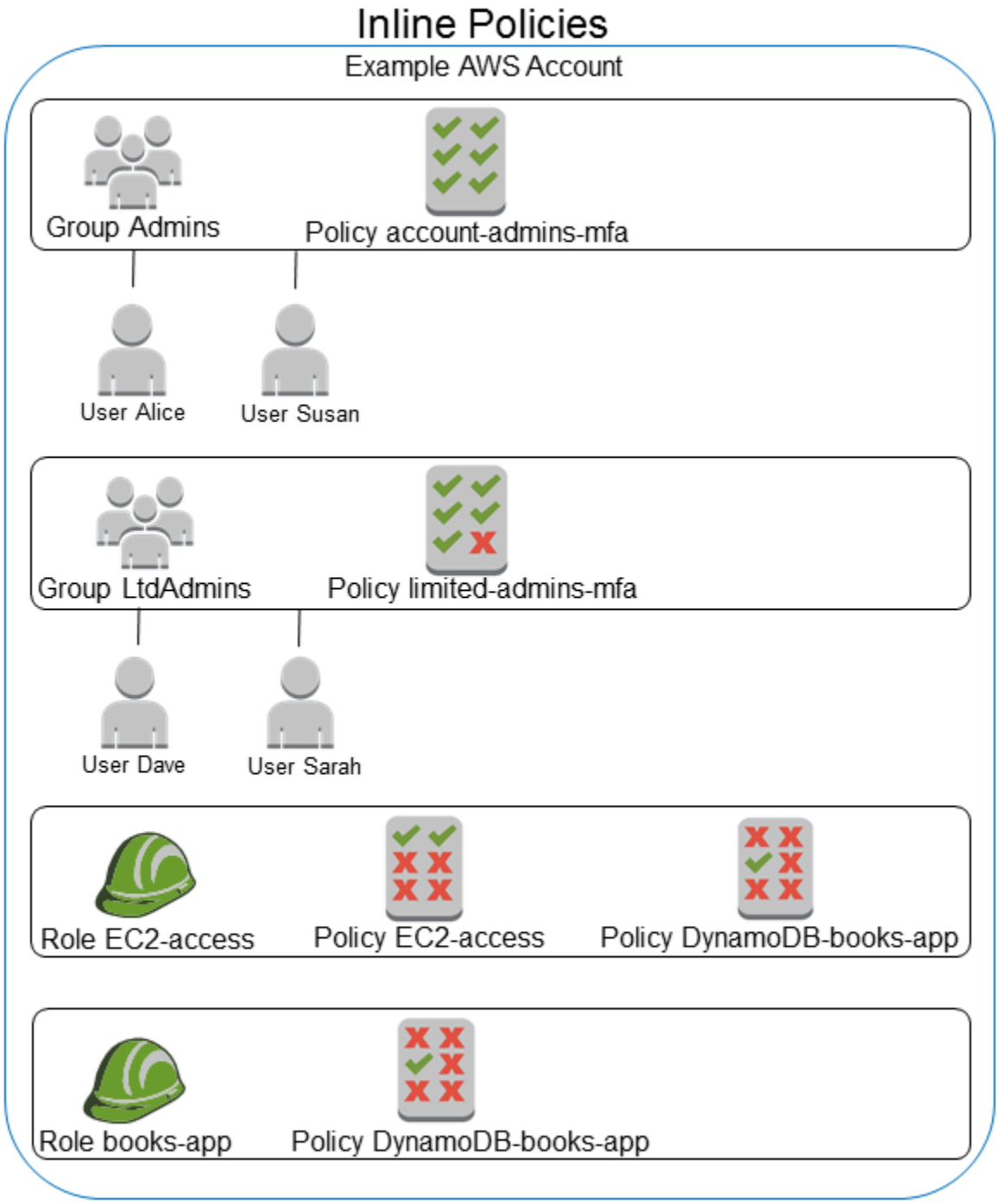
Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien.



Eingebundene Richtlinien

Eine Inline-Richtlinie ist eine Richtlinie, die für eine einzelne IAM-Identität (einen Benutzer, eine Gruppe oder eine Rolle) erstellt worden ist. Inline-Richtlinien sorgen für eine strikte one-to-one Beziehung zwischen einer Richtlinie und einer Identität. Sie werden gelöscht, wenn Sie die Identität löschen. Sie können eine Richtlinie erstellen und sie entweder, wenn Sie die Identität erstellen, oder später in eine Identität einbetten. Wenn eine Richtlinie für mehr als eine Entität gelten könnte, ist es besser, eine verwaltete Richtlinie zu verwenden.

Die eingebundenen Richtlinien sind in der folgenden Abbildung dargestellt. Jede Richtlinie ist unabdingbarer Bestandteil des Benutzers, der Gruppe oder der Rolle. Beachten Sie, dass zwei Rollen dieselbe Richtlinie enthalten (die Richtlinie DynamoDB-books-app), aber sie verwenden keine Richtlinie gemeinsam. Jede Rolle hat ihre eigene Kopie der Richtlinie.



Auswahl zwischen verwalteten und eingebundenen Richtlinien

Berücksichtigen Sie Ihre Anwendungsfälle, wenn Sie zwischen verwalteten und Inline-Richtlinien entscheiden. In den meisten Fällen empfehlen wir, dass Sie verwaltete Richtlinien anstelle von eingebundenen Richtlinien verwenden.

> **Note**
>
> Sie können sowohl verwaltete als auch Inline-Richtlinien zusammen verwenden, um gemeinsame und eindeutige Berechtigungen für eine Prinzipal-Entität zu definieren.

Verwaltete Richtlinien bieten die folgenden Funktionen:

Wiederverwendbarkeit

Eine einzelne verwaltete Richtlinie kann an mehrere Auftraggeber-Entitäten (Benutzer, Gruppen und Rollen) angefügt werden. Sie können eine Bibliothek mit Richtlinien erstellen, die nützliche Berechtigungen für Sie definieren AWS-Konto, und diese Richtlinien dann nach Bedarf an Prinzipalentitäten anhängen.

Zentrale Änderungsverwaltung

Wenn Sie eine verwaltete Richtlinie ändern, wird die Änderung auf alle Auftraggeber-Entitäten angewendet, an die die Richtlinie angefügt ist. Wenn Sie beispielsweise Berechtigungen für eine neue AWS API hinzufügen möchten, können Sie eine vom Kunden verwaltete Richtlinie aktualisieren oder eine AWS verwaltete Richtlinie zuordnen, um die Berechtigung hinzuzufügen. Wenn Sie eine AWS verwaltete Richtlinie verwenden, AWS aktualisiert die Richtlinie. Wenn eine verwaltete Richtlinie aktualisiert wird, werden die Änderungen auf alle Prinzipalentitäten angewendet, denen die verwaltete Richtlinie zugeordnet ist. Um eine Inline-Richtlinie zu ändern, müssen Sie dagegen jede Identität, die die Inline-Richtlinie enthält, einzeln bearbeiten. Wenn beispielsweise eine Gruppe und eine Rolle dieselbe Inline-Richtlinie enthalten, müssen Sie beide Prinzipal-Entitäten individuell bearbeiten, um diese Richtlinie zu ändern.

Versioning und Rollback

Wenn Sie eine kundenverwaltete Richtlinie ändern, wird die vorhandene Richtlinie nicht von der geänderten Richtlinie überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. IAM speichert bis zu fünf Versionen Ihrer vom Kunden verwalteten Richtlinien. Sie können Richtlinienversionen verwenden, um eine frühere Version einer Richtlinie wiederherzustellen, sofern dies erforderlich ist.

> **Note**
>
> Eine Richtlinienversion ist nicht mit dem Richtlinienelement `Version` identisch. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Weitere Informationen zu den Richtlinienversionen finden Sie unter the section called “Versioning von IAM-Richtlinien”. Weitere Informationen zum Richtlinienelement `Version` finden Sie unter IAM-JSON-Richtlinienelemente: Version.

Delegieren der Berechtigungsverwaltung

Sie können Benutzern in Ihren AWS-Konto Richtlinien das Anhängen und Trennen von Richtlinien gestatten und gleichzeitig die Kontrolle über die in diesen Richtlinien definierten Berechtigungen behalten. Sie können einige Benutzer als Administratoren mit vollständigen Rechten bestimmen, d. h. Administratoren, die Richtlinien erstellen, aktualisieren und löschen können. Anschließend können Sie andere Benutzer als Administratoren mit eingeschränkten Rechten bestimmen. Dies bedeutet, dass Administratoren zwar Richtlinien an andere Prinzipal-Entitäten anfügen können, aber nur die Richtlinien, für die Sie ihnen die Berechtigungen zum Anfügen erteilt haben.

Weitere Informationen zum Delegieren der Berechtigungsverwaltung finden Sie unter Steuern des Zugriffs auf Richtlinien.

Größere Zeichenbeschränkungen für Richtlinien

Die maximale Zeichengröße für verwaltete Richtlinien ist größer als die Zeichenbeschränkung für eingebundene Richtlinien. Wenn Sie die Zeichengrößenbeschränkung der eingebundenen Richtlinie erreichen, können Sie weitere IAM-Gruppen erstellen und die verwaltete Richtlinie der Gruppe zuweisen.

Weitere Informationen zu Kontingenten und Limits finden Sie unter IAM und Kontingente AWS STS.

Automatische Updates für AWS verwaltete Richtlinien

AWS verwaltet AWS verwaltete Richtlinien und aktualisiert sie bei Bedarf, um beispielsweise Berechtigungen für neue AWS Dienste hinzuzufügen, ohne dass Sie Änderungen vornehmen müssen. Die Updates werden automatisch auf die Prinzipalentitäten angewendet, denen Sie die AWS verwaltete Richtlinie angehängt haben.

Verwenden von eingebundenen Richtlinien

Inline-Richtlinien sind nützlich, wenn Sie eine strikte one-to-one Beziehung zwischen einer Richtlinie und der Identität, auf die sie angewendet wird, aufrechterhalten möchten. Sie sollten beispielsweise darauf achten, dass die Berechtigungen einer Richtlinie nicht versehentlich einer Identität zugewiesen werden, für die sie nicht vorgesehen sind. Bei Verwendung einer Inline-Richtlinie können die Berechtigungen in der Richtlinie nicht versehentlich der falschen Identität zugeordnet werden. Wenn Sie AWS Management Console zum Löschen dieser Identität verwenden, werden außerdem die in der Identität eingebetteten Richtlinien ebenfalls gelöscht, da sie Teil der Prinzipalentität sind.

Erste Schritte mit verwalteten Richtlinien

Es wird empfohlen, Richtlinien zu verwenden, die die geringsten Rechte gewähren, d. h. nur die für die Durchführung einer Aufgabe erforderlichen Berechtigungen zu gewähren. Die sicherste Methode, die geringste Berechtigung zu erteilen, besteht darin, eine vom Kunden verwaltete Richtlinie mit nur den Berechtigungen zu schreiben, die Ihr Team benötigt. Sie müssen einen Prozess erstellen, damit Ihr Team bei Bedarf weitere Berechtigungen anfordern kann. Es erfordert Zeit und Fachwissen, um von Kunden verwaltete IAM-Richtlinien zu erstellen, die Ihrem Team nur die benötigten Berechtigungen bieten.

Um mit dem Hinzufügen von Berechtigungen zu Ihren IAM-Identitäten (Benutzer, Benutzergruppen und Rollen) zu beginnen, können Sie verwenden. AWS verwaltete Richtlinien AWS verwaltete Richtlinien gewähren keine Berechtigungen mit den geringsten Rechten. Sie müssen das Sicherheitsrisiko berücksichtigen, wenn Sie Ihren Auftraggeber mehr Berechtigungen gewähren, als sie für ihre Arbeit benötigen.

Sie können AWS verwaltete Richtlinien, einschließlich Jobfunktionen, an jede IAM-Identität anhängen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

Um zu den Berechtigungen mit den geringsten Rechten zu wechseln, können Sie AWS Identity and Access Management Access Analyzer ausführen, um die Prinzipale mit AWS verwalteten Richtlinien zu überwachen. Nachdem Sie erfahren haben, welche Berechtigungen sie verwenden, können Sie eine vom Kunden verwaltete Richtlinie schreiben oder generieren, die nur die erforderlichen Berechtigungen für Ihr Team beinhaltet. Das ist weniger sicher, bietet aber mehr Flexibilität, wenn Sie erfahren, wie Ihr Team die Daten verwendet AWS. Weitere Informationen finden Sie unter Generieren von IAM Access Analyzer-Richtlinien.

AWS verwaltete Richtlinien dienen dazu, Berechtigungen für viele gängige Anwendungsfälle bereitzustellen. Weitere Informationen zu AWS verwalteten Richtlinien, die für bestimmte Aufgaben konzipiert sind, finden Sie unterAWS verwaltete Richtlinien für Jobfunktionen.

Eine Liste der AWS verwalteten Richtlinien finden Sie im Referenzhandbuch für AWS verwaltete Richtlinien.

So konvertieren Sie eine eingebundene Richtlinie in eine verwaltete Richtlinie

Wenn Sie eingebundene Richtlinien in Ihrem Konto verwenden, können Sie diese in verwaltete Richtlinien umwandeln. Kopieren Sie hierzu die Richtlinie in eine neue verwaltete Richtlinie. Fügen Sie als Nächstes die neue Richtlinie an die Identität an, die die Inline-Richtlinie enthält. Löschen Sie dann die eingebundene Richtlinie.

So konvertieren Sie eine eingebundene Richtlinie in eine verwaltete 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. Klicken Sie im Navigationsbereich auf Groups (Gruppen), Users (Benutzer) oder Roles (Rollen).
3. Wählen Sie in der Liste den Namen der Gruppe, des Benutzers oder der Rolle aus, deren bzw. dessen Richtlinie Sie entfernen möchten.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wählen Sie für Benutzergruppen den Namen der eingebundenen Richtlinie aus, die Sie entfernen möchten. Wählen Sie für Benutzer und Rollen gegebenenfalls Show ***n*** more (n weitere anzeigen) und den Pfeil neben der Inline-Richtlinie aus, die Sie entfernen möchten.
6. Wählen Sie Copy (Kopieren) aus, um das JSON-Richtliniendokument für die Richtlinie zu kopieren.
7. Wählen Sie im Navigationsbereich Richtlinien.
8. Wählen Sie Create policy (Richtlinie erstellen) und anschließend die Option JSON aus.
9. Ersetzen Sie den vorhandenen Text durch den Text der JSON-Richtlinie und wählen Sie dann Next (Weiter) aus.
10. Geben Sie einen Namen und eine optionale Beschreibung für Ihre Richtlinie ein und wählen Sie dann Create Policy (Richtlinie erstellen) aus.
11. Wählen Sie im Navigationsbereich Groups (Gruppen), Users (Benutzer) oder Roles (Rollen) und dann den Namen der Gruppe, des Benutzers oder der Rolle aus, die die Richtlinie enthält, die Sie entfernen möchten.

12. Wählen Sie die Registerkarte Permissions (Berechtigungen) und anschließend die Option Add Permissions (Berechtigungen hinzufügen).
13. Aktivieren Sie bei Benutzergruppen das Kontrollkästchen neben dem Namen der neuen Richtlinie, wählen Sie Berechtigungen hinzufügen und dann Richtlinie anfügen. Wählen Sie für Benutzer oder Rollen Add permissions (Berechtigungen hinzufügen). Wählen Sie auf der nächsten Seite Attach existing policies directly (Vorhandene Richtlinien direkt anfügen) aus, aktivieren Sie das Kontrollkästchen neben dem Namen der neuen Richtlinie und wählen Sie dann Next (Weiter) und Add permissions (Berechtigungen hinzufügen) aus.

    Die Seite Summary (Übersicht) für die Gruppe, den Benutzer oder die Rolle wird erneut angezeigt.
14. Aktivieren Sie das Kontrollkästchen neben der Inline-Richtlinie, die Sie entfernen möchten, und wählen Sie Remove (Entfernen).

Veraltete verwaltete Richtlinien AWS

AWS Stellt verwaltete Richtlinien bereit, um die Zuweisung von Berechtigungen zu vereinfachen. Dabei handelt es sich um vordefinierte Richtlinien, die Ihren IAM-Benutzern, -Gruppen und -Rollen zugewiesen werden können.

Manchmal AWS muss einer bestehenden Richtlinie eine neue Berechtigung hinzugefügt werden, z. B. wenn ein neuer Dienst eingeführt wird. Durch das Hinzufügen einer neuen Berechtigung zu einer vorhandenen Richtlinie werden keine Eigenschaften oder Fähigkeiten beeinflusst oder gelöscht.

AWS Möglicherweise entscheiden Sie sich jedoch für die Erstellung einer neuen Richtlinie, wenn die erforderlichen Änderungen Auswirkungen auf Kunden haben könnten, wenn sie auf eine bestehende Richtlinie angewendet würden. Das Entfernen von Berechtigungen aus einer vorhandenen Richtlinie könnte die Berechtigungen der IAM-Entitäten oder Anwendungen, die von dieser Richtlinie abhängig sind, aufheben und kritische Vorgänge unterbrechen.

Wenn eine solche Änderung erforderlich ist, wird daher eine völlig neue Richtlinie mit den erforderlichen Änderungen AWS erstellt und den Kunden zur Verfügung gestellt. Die alte Richtlinie wird dann als veraltet gekennzeichnet. Eine veraltete, verwaltete Richtlinie wird mit einem Warnsymbol in der Liste Policies in der IAM-Konsole angezeigt.

Eine veraltete Richtlinie hat folgende Merkmale:

- Sie ist unverändert für alle gegenwärtig angefügten Benutzer, Gruppen und Rollen funktionsfähig. Alles funktioniert normal.

- Sie kann nicht neuen Benutzern, Gruppen oder Rollen angefügt werden. Wenn Sie sie von einer gegenwärtigen Entität trennen, können Sie sie nicht wieder anfügen.
- Nachdem Sie sie von allen aktuellen Entitäten getrennt haben, ist sie nicht mehr sichtbar und kann nicht mehr verwendet werden.

Wenn für einen Benutzer, eine Gruppe oder Rolle eine Richtlinie erforderlich ist, müssen Sie eine neue Richtlinie anfügen. Wenn Sie feststellen, dass eine Richtlinie veraltet ist, empfehlen wir, dass Sie sofort vorsehen, allen Benutzern Gruppen und Rollen die Ersatzrichtlinie anzufügen und sie anschließend von der veralteten Richtlinie zu trennen. Die weitere Verwendung einer veralteten Richtlinie kann Risiken bergen, die nur durch den Wechsel zur Ersatz-Richtlinie zu vermeiden sind.

Richten Sie mithilfe von Datenperimetern Richtlinien für Genehmigungen ein

Leitplanken an den Datengrenzen dienen als ständig verfügbare Grenzen, um Ihre Daten über eine Vielzahl von Konten und Ressourcen hinweg zu schützen. AWS Datenperimeter folgen den bewährten IAM-Sicherheitsmethoden, um Schutzmaßnahmen für mehrere Konten einzurichten. Diese organisationsweiten Richtlinien für Berechtigungen ersetzen nicht Ihre bestehenden, detaillierten Zugriffskontrollen. Stattdessen dienen sie als grobkörnige Zugriffskontrollen, die zur Verbesserung Ihrer Sicherheitsstrategie beitragen, indem sie sicherstellen, dass Benutzer, Rollen und Ressourcen eine Reihe definierter Sicherheitsstandards einhalten.

Bei einem Datenperimeter handelt es sich um eine Reihe von Schutzplanken in Ihrer AWS Umgebung, die sicherstellen, dass nur Ihre vertrauenswürdigen Identitäten auf vertrauenswürdige Ressourcen von erwarteten Netzwerken zugreifen.

- Vertrauenswürdige Identitäten: Principals (IAM-Rollen oder Benutzer) in Ihren AWS Konten und Diensten, die in Ihrem Namen handeln. AWS
- Vertrauenswürdige Ressourcen: Ressourcen, die Ihren AWS Konten oder AWS Diensten gehören, die in Ihrem Namen handeln.
- Erwartete Netzwerke: Ihre lokalen Rechenzentren und Virtual Private Clouds (VPCs) oder Netzwerke von AWS Diensten, die in Ihrem Namen agieren.

> **Note**
>
> In einigen Fällen müssen Sie möglicherweise Ihren Datenperimeter erweitern, um auch den Zugriff durch Ihre vertrauenswürdigen Geschäftspartner einzubeziehen. Sie sollten alle beabsichtigten Datenzugriffsmuster berücksichtigen, wenn Sie eine Definition von vertrauenswürdigen Identitäten, vertrauenswürdigen Ressourcen und erwarteten Netzwerken erstellen, die speziell auf Ihr Unternehmen und Ihre Nutzung zugeschnitten sind. AWS-Services

Datenperimeterkontrollen sollten wie jede andere Sicherheitskontrolle im Rahmen des Informationssicherheits- und Risikomanagementprogramms behandelt werden. Das bedeutet, dass Sie eine Bedrohungsanalyse durchführen sollten, um potenzielle Risiken in Ihrer Cloud-Umgebung zu identifizieren, und dann auf der Grundlage Ihrer eigenen Risikoakzeptanzkriterien geeignete Kontrollen für den Datenperimeter auswählen und implementieren sollten. Um den iterativen, risikobasierten Ansatz zur Implementierung von Datenperimetern besser nachvollziehen zu können, müssen Sie sich darüber im Klaren sein, welche Sicherheitsrisiken und Bedrohungsvektoren durch Datenperimeterkontrollen angegangen werden und welche Sicherheitsprioritäten Sie haben.

Kontrollen am Datenperimeter

Grobkörnige Kontrollen am Datenperimeter helfen Ihnen dabei, sechs unterschiedliche Sicherheitsziele in drei Datenperimetern zu erreichen, indem Sie verschiedene Kombinationen von und Zustandsschlüsseln implementieren. Richtlinientypen

Perimeter	Ziel der Kontrolle	Die Verwendung von	Angewendet auf	Kontextschlüssel für globale Bedingungen
Identität	Nur vertrauen swürdige Identität en können auf meine Ressourcen zugreifen	Ressource nbasierte Richtlinie	Ressourcen	aws: ID Principal Org war: Principal OrgPaths war: Principal Account

Perimeter	Ziel der Kontrolle	Die Verwendung von	Angewendet auf	Kontextschlüssel für globale Bedingungen
	In meinem Netzwerk sind nur vertrauen swürdige Identitäten erlaubt	VPC-Endpu nktrichtlinie	Network (Netzwerk)	war: Principal IsAwsService
Ressourcen	Ihre Identität en können nur auf vertrauen swürdige Ressourcen zugreifen	SCP	Identitäten	aws: ID ResourceOrg war: ResourceO rgPaths war: ResourceA ccount
	Von Ihrem Netzwerk aus kann nur auf vertrauen swürdige Ressourcen zugegriffen werden	VPC-Endpu nktrichtlinie	Network (Netzwerk)	
Network (Netzwerk)	Ihre Identität en können nur von erwarteten Netzwerken aus auf Ressourcen zugreifen	SCP	Identitäten	als: SourceIp war: SourceVpc war: SourceVpce AWS: über AWSService war: Principal IsAwsService

Perimeter	Ziel der Kontrolle	Die Verwendung von	Angewendet auf	Kontextschlüssel für globale Bedingungen
	Auf Ihre Ressourcen kann nur von den erwarteten Netzwerken aus zugegriffen werden	Ressourcenbasierte Richtlinie	Ressourcen	

Sie können sich Datenperimeter so vorstellen, dass sie eine feste Grenze zwischen Ihren Daten bilden, um unbeabsichtigte Zugriffsmuster zu verhindern. Datenperimeter können zwar weitreichenden unbeabsichtigten Zugriff verhindern, Sie müssen jedoch dennoch detaillierte Entscheidungen zur Zugriffskontrolle treffen. Die Einrichtung eines Datenperimeters mindert nicht die Notwendigkeit, die Berechtigungen mithilfe von Tools wie IAM Access Analyzer kontinuierlich zu optimieren, als Teil Ihrer Reise zu den geringsten Rechten.

Identitätsperimeter

Ein Identitätsperimeter besteht aus einer Reihe von groben präventiven Zugriffskontrollen, die sicherstellen, dass nur vertrauenswürdige Identitäten auf Ihre Ressourcen zugreifen können und dass nur vertrauenswürdige Identitäten in Ihrem Netzwerk zugelassen werden. Zu den vertrauenswürdigen Identitäten gehören Principals (Rollen oder Benutzer) in Ihren Konten und Diensten, die in Ihrem Namen handeln. AWS AWS Alle anderen Identitäten gelten als nicht vertrauenswürdig und werden vom Identitätsperimeter verhindert, sofern keine ausdrückliche Ausnahme gewährt wird.

Die folgenden globalen Bedingungsschlüssel helfen dabei, Kontrollen am Identitätsrand durchzusetzen. Verwenden Sie diese Schlüssel in ressourcenbasierten Richtlinien, um den Zugriff auf Ressourcen einzuschränken, oder in VPC-Endpunktrichtlinien, um den Zugriff auf Ihre Netzwerke einzuschränken.

- aws: PrincipalOrg ID— Mit diesem Bedingungsschlüssel können Sie sicherstellen, dass die IAM-Prinzipale, die die Anfrage stellen, der angegebenen Organisation angehören. AWS Organizations
- aws: PrincipalOrg Pfade— Mit diesem Bedingungsschlüssel können Sie sicherstellen, dass der IAM-Benutzer, die IAM-Rolle, der Verbundbenutzer oder A Root-Benutzer des AWS-Kontos ,

der die Anfrage gestellt hat, der angegebenen Organisationseinheit (OU) in angehört. AWS Organizations

- als: PrincipalAccount— Mit diesem Bedingungsschlüssel können Sie sicherstellen, dass auf Ressourcen nur über das Hauptkonto zugegriffen werden kann, das Sie in der Richtlinie angeben.
- als: PrincipalIs AWSServiceund aws: SourceOrg ID (alternativ aws: SourceOrg Pfade undaws: SourceAccount) — Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass AWS-Service Prinzipale, wenn sie auf Ihre Ressourcen zugreifen, dies nur im Namen einer Ressource in der angegebenen Organisation, Organisationseinheit oder einem Konto in tun. AWS Organizations

Weitere Informationen finden Sie unter Einrichtung eines Datenperimeters unter AWS: Nur vertrauenswürdigen Identitäten den Zugriff auf Unternehmensdaten erlauben.

Ressourcenperimeter

Ein Ressourcenperimeter besteht aus einer Reihe von groben präventiven Zugriffskontrollen, die sicherstellen, dass Ihre Identitäten nur auf vertrauenswürdige Ressourcen zugreifen können und dass nur vertrauenswürdige Ressourcen von Ihrem Netzwerk aus zugänglich sind. Zu den vertrauenswürdigen Ressourcen gehören Ressourcen, die Ihren AWS Konten oder Diensten gehören, die in Ihrem Namen handeln. AWS

Die folgenden globalen Bedingungsschlüssel helfen dabei, die Kontrolle der Ressourcengrenzen durchzusetzen. Verwenden Sie diese Schlüssel in Service Control Policies (SCPs), um einzuschränken, auf welche Ressourcen Ihre Identitäten zugreifen können, oder in VPC-Endpunktrichtlinien, um einzuschränken, auf welche Ressourcen von Ihren Netzwerken aus zugegriffen werden kann.

- aws: ResourceOrg ID— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass die Ressource, auf die zugegriffen wird, der angegebenen Organisation in gehört. AWS Organizations
- aws: ResourceOrg Pfade— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass die Ressource, auf die zugegriffen wird, zur angegebenen Organisationseinheit (OU) in gehört AWS Organizations.
- aws: ResourceAccount— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass die Ressource, auf die zugegriffen wird, zu dem angegebenen Konto in gehört AWS Organizations.

In einigen Fällen müssen Sie möglicherweise den Zugriff auf AWS eigene Ressourcen gewähren, d. h. auf Ressourcen, die nicht zu Ihrer Organisation gehören und auf die Ihre Prinzipale oder in Ihrem Namen handelnde AWS Dienste zugreifen. Weitere Informationen zu diesen Szenarien finden Sie unter Einrichtung eines Datenperimeters unter AWS: Nur vertrauenswürdige Ressourcen aus meiner Organisation zulassen.

Netzwerkperimeter

Bei einem Netzwerkperimeter handelt es sich um eine Reihe grober präventiver Zugriffskontrollen, die sicherstellen, dass Ihre Identitäten nur über erwartete Netzwerke auf Ressourcen zugreifen können und dass auf Ihre Ressourcen nur von erwarteten Netzwerken aus zugegriffen werden kann. Zu den erwarteten Netzwerken gehören Ihre lokalen Rechenzentren und Virtual Private Clouds (VPCs) sowie Netzwerke von Diensten, die in Ihrem Namen agieren. AWS

Die folgenden globalen Bedingungsschlüssel helfen dabei, Netzwerkperimeterkontrollen durchzusetzen. Verwenden Sie diese Schlüssel in Service Control Policies (SCPs), um Netzwerke einzuschränken, von denen aus Ihre Identitäten kommunizieren können, oder in ressourcenbasierten Richtlinien, um den Ressourcenzugriff auf erwartete Netzwerke einzuschränken.

- aws: SourceIp— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass die IP-Adresse des Anfragenden innerhalb eines bestimmten IP-Bereichs liegt.
- aws: SourceVpc— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass der VPC-Endpunkt, den die Anfrage durchläuft, zur angegebenen VPC gehört.
- als: SourceVpce— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass die Anfrage den angegebenen VPC-Endpunkt durchläuft.
- AWS: via AWSService— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass Anfragen im Namen Ihres Auftraggebers über Forward Access Sessions (FAS) (FAS) gestellt werden AWS-Services können.
- als: PrincipalIs AWSService— Sie können diesen Bedingungsschlüssel verwenden, um sicherzustellen, dass AWS-Services Sie über AWS Dienstprinzipale

Es gibt weitere Szenarien, in denen Sie den Zugriff AWS-Services auf Ihre Ressourcen von außerhalb Ihres Netzwerks zulassen müssen. Weitere Informationen finden Sie unter Einrichtung eines Datenperimeters unter AWS: Zugriff auf Unternehmensdaten nur von erwarteten Netzwerken aus zulassen.

Ressourcen, um mehr über Datenperimeter zu erfahren

Die folgenden Ressourcen können Ihnen helfen, mehr über Datenperimeter in allen Bereichen zu erfahren. AWS

- Datenperimeter auf AWS — Erfahren Sie mehr über Datenperimeter und ihre Vorteile und Anwendungsfälle.
- Whitepaper: Building a Data Perimeter on AWS — In diesem paper werden die bewährten Methoden und verfügbaren Dienste für die Einrichtung eines Perimeters rund um Ihre Identitäten, Ressourcen und Netzwerke in beschrieben. AWS
- Webinar: Aufbau eines Datenperimeters in AWS — Erfahren Sie, wo und wie Sie Datenperimeterkontrollen auf der Grundlage verschiedener Risikoszenarien implementieren können.
- Blogbeitragsserie: Einrichtung eines Datenperimeters am AWS — In diesen Blogbeiträgen finden Sie präskriptive Leitlinien für die Einrichtung Ihres Datenperimeters in großem Maßstab, einschließlich wichtiger Sicherheits- und Implementierungsaspekte.
- Beispiele für Datenperimeterrichtlinien — Dieses GitHub Repository enthält Beispielrichtlinien, die einige gängige Muster behandeln, um Sie bei der Implementierung eines Datenperimeters zu unterstützen. AWS
- Data Perimeter Helper — Dieses Tool hilft Ihnen, die Auswirkungen Ihrer Datenperimeterkontrollen zu entwerfen und vorherzusehen, indem es die Zugriffsaktivitäten in Ihren Protokollen analysiert. AWS CloudTrail

Berechtigungsgrenzen für IAM-Entitäten

AWS unterstützt Berechtigungsgrenzen für IAM-Entitäten (Benutzer oder Rollen). Eine Berechtigungsgrenze ist ein erweitertes Feature, bei der Sie eine verwaltete Richtlinie verwenden, um die maximalen Berechtigungen festzulegen, die eine identitätsbasierte Richtlinie einer IAM-Entität gewähren kann. Durch eine Berechtigungsgrenze kann eine Entität nur die Aktionen durchführen, die sowohl von den identitätsbasierten Richtlinien als auch den Berechtigungsgrenzen erlaubt werden.

Weitere Informationen zu Richtlinientypen finden Sie unter Richtlinientypen.

> **Important**
>
> Verwenden Sie keine ressourcenbasierten Richtlinienanweisungen, die ein `NotPrincipal`-Richtlinienelement mit einer `Deny`-Wirkung für IAM-Benutzer oder -Rollen enthalten,

denen eine Richtlinie mit Berechtigungsgrenzen angefügt ist. Das `NotPrincipal`-Element mit `Deny`-Wirkung lehnt immer jeden IAM-Prinzipal ab, an den eine Richtlinie zur Berechtigungsgrenze angefügt ist, unabhängig von den im `NotPrincipal`-Element angegebenen Werten. Dies führt dazu, dass einige IAM-Benutzer oder -Rollen, die andernfalls Zugriff auf die Ressource hätten, den Zugriff verlieren. Wir empfehlen Ihnen, Ihre ressourcenbasierten Richtlinien dahingehend zu ändern, dass Sie den Bedingungsoperator `ArnNotEquals` mit dem `aws:PrincipalArn`-Kontextschlüssel verwenden, um den Zugriff zu begrenzen, anstatt das `NotPrincipal`-Element. Weitere Informationen zum `NotPrincipal`-Element finden Sie unter AWS JSON-Richtlinienelemente: NotPrincipal.

Sie können eine AWS verwaltete Richtlinie oder eine vom Kunden verwaltete Richtlinie verwenden, um die Grenze für eine IAM-Entität (Benutzer oder Rolle) festzulegen. Diese Richtlinie begrenzt die maximalen Berechtigungen für den Benutzer oder die Rolle.

Nehmen wir beispielsweise an, dass der angegebene IAM-Benutzer nur Amazon S3, Amazon und Amazon CloudWatch EC2 verwalten darf. `ShirleyRodriguez` Um diese Regel durchzusetzen, können Sie die folgende Richtlinie verwenden, um die Berechtigungsgrenze für den Benutzer `ShirleyRodriguez` festzulegen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie eine Richtlinie verwenden, um die Berechtigungsgrenze für einen Benutzer festzulegen, schränkt sie die Berechtigungen des Benutzers ein, erteilt aber selbst keine Berechtigungen. In diesem Beispiel legt die Richtlinie die maximalen Berechtigungen für alle Operationen in Amazon S3 und Amazon EC2 fest. `ShirleyRodriguez` CloudWatch Shirley kann niemals Operationen in einem

anderen Service durchführen, einschließlich IAM, selbst wenn sie eine Berechtigungsrichtlinie hat, die dies erlaubt. Sie können z. B. dem Benutzer `ShirleyRodriguez` die folgende Richtlinie hinzufügen:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Diese Richtlinie erlaubt das Erstellen eines Benutzers in IAM. Wenn Sie diese Berechtigungsrichtlinie dem Benutzer `ShirleyRodriguez` anfügen und Shirley versucht, einen Benutzer zu erstellen, schlägt die Operation fehl. Sie schlägt fehl, weil die Berechtigungsgrenze den `iam:CreateUser`-Vorgang nicht zulässt. Angesichts dieser beiden Richtlinien hat Shirley keine Berechtigung, Operationen in AWS durchzuführen. Sie müssen eine andere Berechtigungsrichtlinie hinzufügen, um Aktionen in anderen Diensten zuzulassen, z. B. Amazon S3. Alternativ können Sie die Berechtigungsgrenze so aktualisieren, dass sie einen Benutzer in IAM erstellen kann.

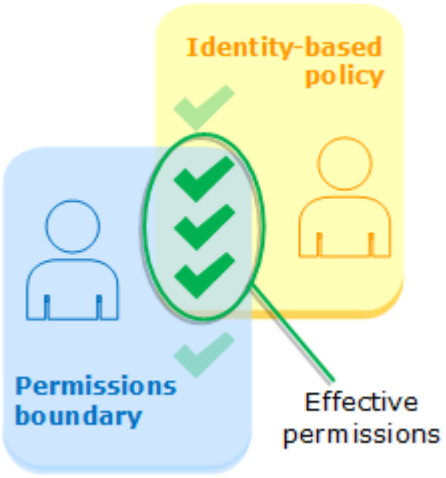
Auswerten von effektiven Berechtigungen mit Grenzen

Die Berechtigungsgrenze für eine IAM-Entität (Benutzer oder Rolle) legt die maximalen Berechtigungen fest, die die Entität haben kann. Dies kann die effektiven Berechtigungen für diesen Benutzer oder diese Rolle ändern. Die effektiven Berechtigungen für eine Entität sind die Berechtigungen, die von allen Richtlinien gewährt werden, die den Benutzer oder die Rolle betreffen. Innerhalb eines Kontos können sich identitätsbasierte Richtlinien, ressourcenbasierte Richtlinien, Berechtigungsgrenzen, Organizations-SCPs oder Sitzungsrichtlinien auf Berechtigungen für eine Entität auswirken. Weitere Informationen zu den unterschiedlichen Richtlinientypen finden Sie unter Berechtigungen und Richtlinien in IAM.

Wenn eine dieser Richtlinientypen explizit den Zugriff für eine Operation verweigert, dann wird die Anforderung abgelehnt. Die Berechtigungen, die einer Entität durch mehrere Berechtigungstypen gewährt werden, sind komplexer. Weitere Informationen darüber, wie Richtlinien AWS bewertet werden, finden Sie unter. Auswertungslogik für Richtlinien

Identitätsbasierte Richtlinien mit Grenzen – Identitätsbasierte Richtlinien sind verwaltete oder eingebundene Richtlinien, die einem Benutzer, einer Gruppe von Benutzern oder einer Rolle zugewiesen sind. Identitätsbasierte Richtlinien erteilen die Berechtigung für die Entität,

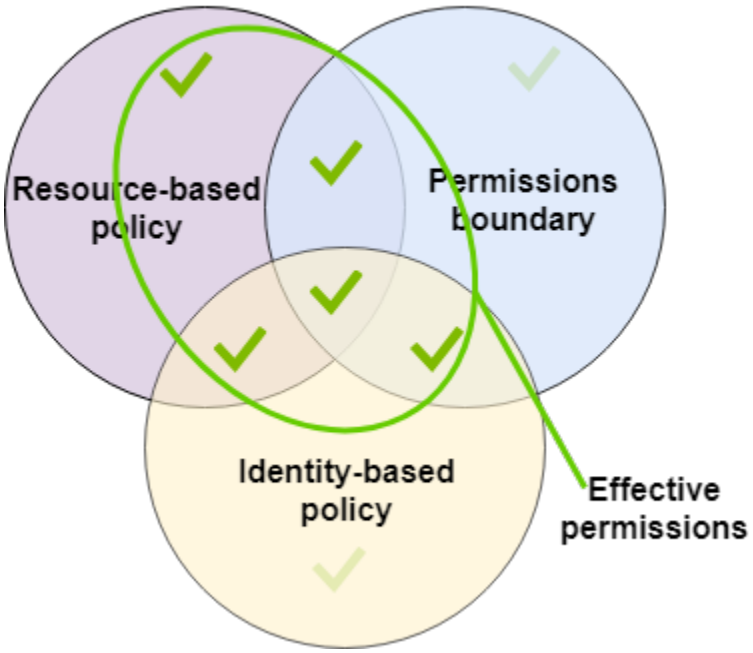
Berechtigungsgrenzen beschränken diese Berechtigungen. Effektive Berechtigungen ergeben sich aus der Überlschneidung der beiden Richtlinientypen. Eine explizite Zugriffsverweigerung in einer der beiden Richtlinien überschreibt die Zugriffserlaubnis.



Ressourcenbasierte Richtlinien – Ressourcenbasierte Richtlinien steuern, wie der angegebene Auftraggeber auf die Ressource zugreifen kann, der die Richtlinie angefügt ist.

Ressourcenbasierte Richtlinien für IAM-Benutzer

> Innerhalb desselben Kontos sind ressourcenbasierte Richtlinien, die einem IAM-Benutzer-ARN (der keine Verbundbenutzersitzung ist) Berechtigungen erteilen, nicht durch eine implizite Verweigerung in einer identitätsbasierten Richtlinie oder Berechtigungsgrenze beschränkt.



Ressourcenbasierte Richtlinien für IAM-Rollen

IAM-Rolle – Ressourcenbasierte Richtlinien, die Berechtigungen für einen IAM-Rollen-ARN erteilen, sind durch eine implizite Verweigerung in einer Berechtigungsgrenze oder einer Sitzungsrichtlinie beschränkt.

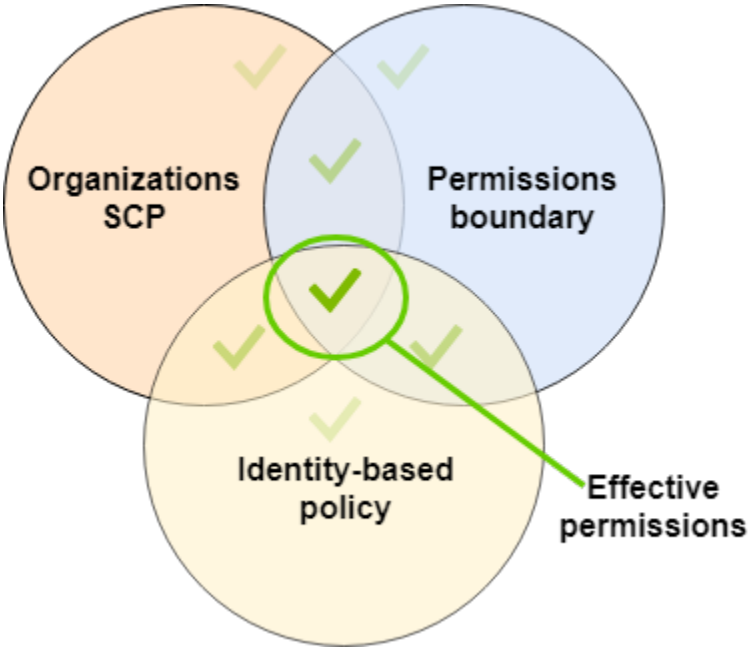
IAM-Rollensitzung – Innerhalb desselben Kontos gewähren ressourcenbasierte Richtlinien, die einem IAM-Rollensitzungs-ARN Berechtigungen erteilen, auch direkt Berechtigungen für die angenommene Rollensitzung. Berechtigungen, die direkt für eine Sitzung gewährt werden, sind nicht durch eine implizite Verweigerung in einer identitätsbasierten Richtlinie, einer Berechtigungsgrenze oder einer Sitzungsrichtlinie beschränkt. Wenn Sie eine Rolle übernehmen und eine Anfrage stellen, ist der Prinzipal, der die Anfrage stellt, der ARN der IAM-Rollensitzung und nicht der ARN der Rolle selbst.

Ressourcenbasierte Richtlinien für IAM-Verbundbenutzersitzungen

IAM-Verbundbenutzersitzung – Eine IAM-Verbundbenutzersitzung ist eine Sitzung, die durch Aufruf von `GetFederationToken` erstellt wurde. Wenn ein Verbundbenutzer eine Anfrage stellt, ist der Prinzipal, der die Anfrage stellt, der ARN des Verbundbenutzers und nicht der ARN des IAM-Benutzers, der den Verbund erstellt hat. Innerhalb desselben Kontos erteilen ressourcenbasierte Richtlinien, die einem Verbundbenutzer-ARN Berechtigungen gewähren, der Sitzung direkt Berechtigungen. Berechtigungen, die direkt für eine Sitzung gewährt werden, sind nicht durch eine implizite Verweigerung in einer identitätsbasierten Richtlinie, einer Berechtigungsgrenze oder einer Sitzungsrichtlinie beschränkt.

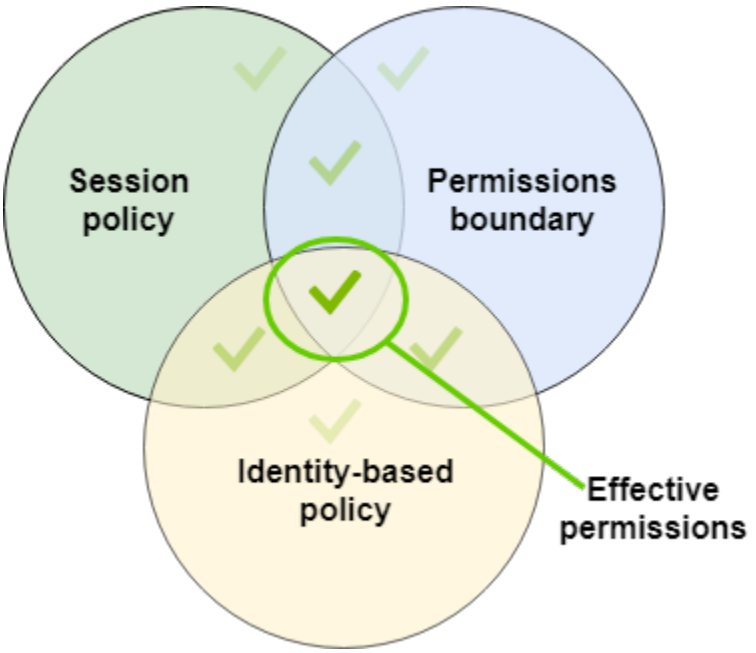
Wenn jedoch eine ressourcenbasierte Richtlinie dem ARN des IAM-Benutzers, der den Verbund erstellt hat, die Berechtigung erteilt, werden Anfragen des Verbundbenutzers während der Sitzung durch eine implizite Verweigerung in einer Berechtigungsgrenze oder Sitzungsrichtlinie eingeschränkt.

Organisationen-SCPs – SCPs werden auf ein gesamtes AWS-Konto angewendet. Sie schränken die Berechtigungen für jede Anforderung ein, die von einem Auftraggeber innerhalb des Kontos gesendet wurden. eine IAM-Entität (Benutzer oder Rolle) kann eine Anforderung stellen, die durch eine SCP, eine Berechtigungsgrenze und eine identitätsbasierte Richtlinie beeinflusst wird. In diesem Fall wird die Anforderung nur gewährt, wenn alle drei Richtlinientypen sie zulassen. Die effektiven Berechtigungen ergeben sich aus der Überschneidung der drei Richtlinientypen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft.



Sie können erfahren ob Ihr Konto als Mitgliedskonto einer Organisation in AWS Organizations eingerichtet ist. Eine SCP hat möglicherweise Auswirkungen auf Mitglieder einer Organisation. Um diese Daten mithilfe des AWS CLI Befehls oder der AWS API-Operation anzuzeigen, müssen Sie über die Berechtigungen für die `organizations:DescribeOrganization` Aktion für Ihre Organisationseinheit verfügen. Sie müssen über zusätzliche Berechtigungen verfügen, um die Operation in der Organizations-Konsole auszuführen. Um zu erfahren, ob ein SCP den Zugriff auf eine bestimmte Anfrage verweigert, oder um Ihre aktuellen Berechtigungen zu ändern, wenden Sie sich an Ihren AWS Organizations Administrator.

Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer programmgesteuert erstellen. Die Berechtigungen für eine Sitzung stammen aus der IAM-Entität (Benutzer oder Rolle), die verwendet wird, um die Sitzung zu erstellen, und von der Sitzungsrichtlinie. Die identitätsbasierten Richtlinienberechtigungen der Entität werden von der Sitzungsrichtlinie und der Berechtigungsgrenze eingeschränkt. Die effektiven Berechtigungen für diese Gruppe von Richtlinientypen ergeben sich aus der Überschneidung aller drei Richtlinientypen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen zu Sitzungsrichtlinien finden Sie unter Sitzungsrichtlinien.



Verantwortung mit Hilfe von Berechtigungsgrenzen an andere delegieren

Sie können Berechtigungsgrenzen verwenden, um Berechtigungsverwaltungsaufgaben, wie das Erstellen von Benutzern, an IAM-Benutzer in Ihrem Konto zu delegieren. Dies erlaubt es anderen, Aufgaben in Ihrem Namen innerhalb einer bestimmten Berechtigungsgrenze auszuführen.

Nehmen wir zum Beispiel an, dass María die Administratorin der X-Company ist. AWS-Konto Sie möchte die Aufgaben der Benutzererstellung an Zhang delegieren. Sie muss jedoch sicherstellen, dass Zhang Benutzer erstellt, die sich an die folgenden Unternehmensregeln halten:

- Benutzer können IAM nicht verwenden, um Benutzer, Gruppen, Rollen oder Richtlinien zu erstellen oder zu verwalten.
- Benutzer erhalten keinen Zugriff auf den Amazon S3 `logs`-Bucket und keinen Zugriff auf die `i-1234567890abcdef0` Amazon EC2-Instance.
- Benutzer können ihre eigenen Begrenzungsrichtlinien nicht entfernen.

Um diese Regeln durchzusetzen, führt María die folgenden Aufgaben durch, die im Folgenden näher erläutert werden:

1. María erstellt die verwaltete Richtlinie `XCompanyBoundaries` als Berechtigungsgrenze für alle neuen Benutzer im Konto.

2. María erstellt die verwaltete Richtlinie `DelegatedUserBoundary` und weist sie als Berechtigungsgrenze für Zhang zu. Maria notiert den ARN ihres Administratorbenutzers und verwendet ihn in der Richtlinie, um zu verhindern, dass Zhang darauf zugreifen kann.
3. María erstellt die verwaltete Richtlinie `DelegatedUserPermissions` und fügt sie als Berechtigungsrichtlinie für Zhang hinzu.
4. María informiert Zhang über seine neuen Aufgaben und Grenzen.

Aufgabe 1: María muss zuerst eine verwaltete Richtlinie erstellen, um die Grenzen für die neuen Benutzer festzulegen. María wird es Zhang erlauben, den Benutzern die erforderlichen Rechte zu geben, aber sie möchte, dass diese Benutzer eingeschränkt werden. Zu diesem Zweck erstellt sie die folgende kundenverwaltete Richtlinie mit dem Namen `XCompanyBoundaries`. Diese Richtlinie gewährt die folgenden Aktionen:

- Ermöglicht den Benutzern den vollständigen Zugriff auf mehrere Services
- Ermöglicht eingeschränkten selbstverwaltenden Zugriff in der IAM-Konsole. Das bedeutet, dass sie ihr Passwort ändern können, nachdem sie sich bei der Konsole angemeldet haben. Sie können ihr anfängliches Passwort nicht festlegen. Um dies zu ermöglichen, fügen Sie die Aktion `"*LoginProfile"` zur Anweisung `AllowManageOwnPasswordAndAccessKeys` hinzu.
- Verweigert Benutzern den Zugriff auf den Amazon S3 Protokoll-Bucket oder die `i-1234567890abcdef0`-Amazon EC2-Instance

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

Jede Anweisung dient einem anderen Zweck:

1. Die `ServiceBoundaries` Erklärung dieser Richtlinie ermöglicht den vollen Zugriff auf die angegebenen AWS Dienste. Dies bedeutet, dass die Aktionen eines neuen Benutzers in diesen Services nur laut der Berechtigungsrichtlinien begrenzt sind, die dem Benutzer zugeordnet sind.

2. Die `AllowIAMConsoleForCredentials`-Anweisung erlaubt den Zugriff für das Auflisten aller IAM-Benutzer. Dieser Zugriff ist erforderlich, um auf der Seite Users (Benutzer) in der AWS Management Console navigieren zu können. Außerdem können die Passwortanforderungen für das Konto angezeigt werden, was erforderlich ist, wenn Sie Ihr eigenes Passwort ändern.
3. Mit der `AllowManageOwnPasswordAndAccessKeys`-Anweisung können Benutzer nur ihr eigenes Konsolen-Passwort und programmatische Zugriffsschlüssel verwalten. Dies ist wichtig, wenn Zhang oder ein anderer Administrator einem neuen Benutzer eine Berechtigungsrichtlinie mit vollem IAM-Zugriff zuweist. In diesem Fall könnte der Benutzer seine eigenen Berechtigungen oder die anderer Benutzer ändern. Diese Anweisung verhindert, dass dies passiert.
4. Die Anweisung `DenyS3Logs` verweigert explizit den Zugriff auf den `logs`-Bucket.
5. Die Anweisung `DenyEC2Production` verweigert explizit den Zugriff auf die `i-1234567890abcdef0`-Instance.

Aufgabe 2: María möchte Zhang erlauben, alle Benutzer für die X-Company Benutzer zu erstellen, aber nur mit der Berechtigungsgrenze `XCompanyBoundaries`. Sie erstellt die folgende kundenverwaltete Richtlinie mit dem Namen `DelegatedUserBoundary`. Diese Richtlinie definiert die maximale Berechtigungen, die Zhang haben kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
```

```
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
              "iam:ListPolicyVersions",
              "iam:ListRolePolicies",
              "iam:ListSSHPublicKeys",
              "iam:ListServiceSpecificCredentials",
              "iam:ListSigningCertificates",
              "iam:ListUserPolicies",
              "iam:ListUsers",
              "iam:SetDefaultPolicyVersion",
              "iam:SimulateCustomPolicy",
              "iam:SimulatePrincipalPolicy",
              "iam:UpdateGroup",
              "iam:UpdateLoginProfile",
              "iam:UpdateUser"
            ],
```

```
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Jede Anweisung dient einem anderen Zweck:

1. Die Anweisung `CreateOrChangeOnlyWithBoundary` erlaubt es Zhang, IAM-Benutzer zu erstellen, aber nur, wenn er die Richtlinie `XCompanyBoundaries` verwendet, um die Berechtigungsgrenze festzulegen. Diese Anweisung erlaubt es ihm auch, die Berechtigungsgrenze für bestehende Benutzer festzulegen, jedoch nur mit der gleichen Richtlinie. Diese Anweisung schließlich erlaubt Zhang, Berechtigungsrichtlinien für Benutzer mit dieser Berechtigungsgrenze zu verwalten.
2. Die Anweisung `CloudWatchAndOtherIAMTasks` ermöglicht es Zhang, andere Aufgaben der Benutzer-, Gruppen- und Richtlinienverwaltung auszuführen. Dieser verfügt über Berechtigungen zum Zurücksetzen von Passwörtern und Erstellen von Zugriffsschlüsseln für alle IAM-Benutzer, die nicht im `NotResource`-Richtlinienelement aufgeführt sind. Dadurch kann er Benutzern bei Anmeldeproblemen helfen.

3. Die Anweisung `NoBoundaryPolicyEdit` verweigert Zhang Zugriff, um die Richtlinie `XCompanyBoundaries` zu aktualisieren. Er darf keine Richtlinien ändern, die dazu dienen, die Berechtigungsgrenze für sich selbst oder andere Benutzer festzulegen.
4. Die `NoBoundaryUserDelete`-Anweisung verweigert Zhang den Zugriff zum Löschen der Berechtigungsgrenze für ihn oder andere Benutzer.

María weist anschließend die`DelegatedUserBoundary` Richtlinie als Berechtigungsgrenze für den Benutzer Zhang zu.

Aufgabe 3: Da die Berechtigungsgrenze die maximalen Berechtigungen begrenzt, aber selbst keinen Zugriff gewährt, muss Maria eine Berechtigungsrichtlinie für Zhang erstellen. Sie erstellt die folgende Richtlinie mit dem Namen `DelegatedUserPermissions`. Diese Richtlinie definiert die Operationen, die Zhang innerhalb der definierten Grenzen ausführen kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
```

```
    ]
}
```

Jede Anweisung dient einem anderen Zweck:

1. Die `IAM`-Anweisung der Richtlinie gewährt Zhang den vollständigen Zugriff auf IAM. Da seine Berechtigungsgrenze jedoch nur einige IAM-Operationen zulässt, sind seine effektiven IAM-Berechtigungen nur durch seine Berechtigungsgrenze begrenzt.
2. Die `CloudWatchLimited` Aussage ermöglicht es Zhang, fünf Aktionen auszuführen. CloudWatch Seine Rechtegrenze lässt alle Aktionen zu CloudWatch, sodass seine effektiven CloudWatch Berechtigungen nur durch seine Berechtigungsrichtlinie eingeschränkt werden.
3. Mit der `S3BucketContents`-Anweisung kann Zhang den Amazon S3-Bucket `ZhangBucket` auflisten. Allerdings erlaubt seine Berechtigungsgrenze keine Amazon S3-Aktion, so kann er keine S3-Operationen durchführen, unabhängig von seiner Berechtigungsrichtlinie.

> **Note**
>
> Zhangs Richtlinien ermöglichen es ihm, einen Benutzer zu erstellen, der dann auf Amazon S3-Ressourcen zugreifen kann, auf die er nicht zugreifen kann. Durch die Delegation dieser administrativen Maßnahmen vertraut Maria effektiv Zhang mit dem Zugriff auf Amazon S3.

María weist anschließend die `DelegatedUserPermissions`-Richtlinie als Berechtigungsrichtlinie für den Benutzer `Zhang` zu.

Aufgabe 4: Sie gibt Zhang Anweisungen, einen neuen Benutzer zu erstellen. Sie sagt ihm, dass er neue Benutzer mit allen erforderlichen Berechtigungen anlegen kann, aber er muss ihnen die Richtlinie `XCompanyBoundaries` als Berechtigungsgrenze zuweisen.

Zhang führt die folgenden Aufgaben aus:

1. Zhang erstellt einen Benutzer mit der AWS Management Console. Er gibt den Benutzernamen `Nikhil` ein und aktiviert den Konsolenzugriff für den Benutzer. Er deaktiviert das Kontrollkästchen neben Passwortrücksetzung erforderlich, da die oben genannten Richtlinien Benutzern erst dann erlauben, ihr Passwort zu ändern, nachdem sich bei der IAM-Konsole angemeldet haben.

2. Auf der Seite „Berechtigungen festlegen“ wählt Zhang die IAM FullAccess - und ReadOnlyAccessAmazonS3-Berechtigungsrichtlinien aus, die es Nikhil ermöglichen, seine Arbeit zu erledigen.
3. Zhang überspringt den Abschnitt Set permissions boundary (Berechtigungsgrenze festlegen) und vergisst die Anweisungen von María.
4. Zhang überarbeitet die Benutzerdetails und wählt Create user (Benutzer erstellen).

   Die Operation schlägt fehl und der Zugriff wird verweigert. Die Berechtigungsgrenze `DelegatedUserBoundary` von Zhang fordert, dass alle Benutzer, die er erstellt, die Richtlinie `XCompanyBoundaries` als Berechtigungsgrenze verwenden.
5. Zhang kehrt zurück auf die vorherige Seite. Im Abschnitt Set permissions boundary (Berechtigungsgrenze festlegen) wählt er die Richtlinie `XCompanyBoundaries`.
6. Zhang überarbeitet die Benutzerdetails und wählt Create user (Benutzer erstellen).

   Der Benutzer wird erstellt.

Wenn Nikhil sich anmeldet, hat er Zugriff auf IAM und Amazon S3, mit Ausnahme der Operationen, die laut der Berechtigungsgrenze verweigert werden. Zum Beispiel kann er sein eigenes Passwort in IAM ändern, aber keinen anderen Benutzer anlegen oder seine Richtlinien bearbeiten. Nikhil hat schreibgeschützten Zugriff auf Amazon S3.

Wenn jemand dem `logs`-Bucket eine ressourcenbasierte Richtlinie zuordnet, die Nikhil das Hinzufügen eines Objekts in den Bucket gewährt, dann kann immer noch nicht auf den Bucket zuzugreifen. Der Grund ist, dass alle Aktionen im `logs`-Bucket durch seine Berechtigungsgrenze explizit verweigert werden. Eine explizite Zugriffsverweigerung in jedem Richtlinientyp führt dazu, dass eine Anforderung verweigert wird. Wenn jedoch eine ressourcenbasierte Richtlinie, die mit einem Secrets Manager-Geheimnis verknüpft ist, Nikhil erlaubt, die `secretsmanager:GetSecretValue`-Aktion durchzuführen, kann Nikhil das Geheimnis abrufen und entschlüsseln. Der Grund dafür ist, dass Secrets Manager-Operationen von seiner Berechtigungsgrenze nicht explizit verweigert werden, und impliziete Zugriffsverweigerungen in Berechtigungsgrenzen beschränken keine ressourcenbasierte Richtlinien.

Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien

Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. Wenn Sie eine Berechtigungsrichtlinie erstellen, um den Zugriff auf

eine Ressource einzuschränken, können Sie zwischen einer identitätsbasierten Richtlinie und einer ressourcenbasierten Richtlinie wählen.

Identitätsbasierte Richtlinien werden an IAM-Benutzer, -Gruppen oder -Rollen angefügt. Mit diesen Richtlinien können Sie festlegen, welche Aktionen diese Identität durchführen darf (ihre Berechtigungen). Sie können die Richtlinie beispielsweise dem IAM-Benutzer John zuordnen und ihm erlauben, die Aktion Amazon EC2 `RunInstances` auszuführen. Die Richtlinie könnte weiterhin besagen, dass John Elemente aus einer Amazon DynamoDB-Tabelle mit dem Namen `MyCompany` abrufen darf. Sie können auch zulassen, dass John seine eigenen IAM-Sicherheitsanmeldeinformationen verwaltet. Identitätsbasierte Richtlinien können verwaltet oder eingebunden sein.

Ressourcenbasierten Richtlinien sind an eine Ressource angefügt. Sie können beispielsweise ressourcenbasierte Richtlinien an Amazon S3-Buckets, Amazon SQS SQS-Warteschlangen, VPC-Endpunkte, AWS Key Management Service Verschlüsselungsschlüssel und Amazon DynamoDB-Tabellen und -Streams anhängen. Eine Liste der Services, die ressourcenbasierte Richtlinien unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

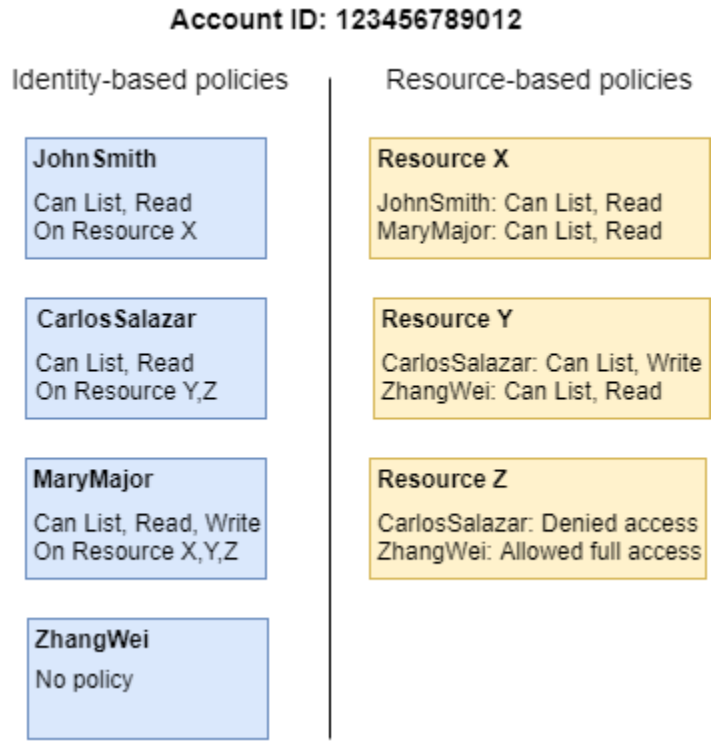
Mit ressourcenbasierten Richtlinien können Sie festlegen, wer Zugriff auf die Ressource hat und welche Aktionen ausgeführt werden können. Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?. Ressourcenbasierten Richtlinien sind nur eingebundene Richtlinien, keine verwalteten Richtlinien.

> **Note**
>
> Ressourcenbasierte Richtlinien unterscheiden sich von Berechtigungen auf Ressourcenebene. Sie können ressourcenbasierte Richtlinien direkt an eine Ressource anfügen, wie in diesem Thema beschrieben. Berechtigungen auf Ressourcenebene beziehen sich auf die Fähigkeit, ARNs zu verwenden, um einzelne Ressourcen in einer Richtlinie anzugeben. Ressourcenbasierte Richtlinien werden nur von einigen Services unterstützt. AWS Eine Liste der Services, die ressourcenbasierte Richtlinien und Berechtigungen auf Ressourcenebene unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

Wie identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien innerhalb desselben Kontos zusammenwirken, erfahren Sie unter Auswerten von Richtlinien in einem einzelnen Konto.

Wie die Richtlinien kontenübergreifend zusammenwirken, erfahren Sie unter Logik für die kontenübergreifende Richtlinienauswertung.

Um ein besseres Verständnis dieser Konzepte zu erhalten, betrachten Sie die folgende Abbildung. Der Administrator des Kontos `123456789012` hat identitätsbasierten Richtlinien an die Benutzer `JohnSmith`, `CarlosSalazar` und `MaryMajor` angefügt. Einige der Aktionen in diesen Richtlinien können für bestimmte Ressourcen ausgeführt werden. Der Benutzer `JohnSmith` kann beispielsweise einige Aktionen für `Resource X` ausführen. Dies ist eine Berechtigung auf Ressourcenebene in einer identitätsbasierten Richtlinie. Der Administrator hat außerdem ressourcenbasierte Richtlinien zu `Resource X`, `Resource Y` und `Resource Z` hinzugefügt. Mithilfe von ressourcenbasierten Richtlinien können Sie festlegen, wer auf diese Ressource zugreifen kann. Die ressourcenbasierte Richtlinie für `Resource X` gewährt beispielsweise den Benutzern `JohnSmith` und `MaryMajor` Lese- und Schreibzugriff auf die Ressource.



Das Kontobeispiel `123456789012` erlaubt den folgenden Benutzern, die aufgeführten Aktionen durchzuführen:

- JohnSmith— John kann Listen- und Leseaktionen für ausführen. `Resource X` Er erhält diese Berechtigung über die identitätsbasierte Richtlinie seines Benutzers sowie die ressourcenbasierte Richtlinien von `Resource X`.

- CarlosSalazar— Carlos kann Listen-, Lese- und Schreibaktionen ausführen`Resource Y`, hat aber keinen Zugriff darauf`Resource Z`. Die identitätsbasierte Richtlinie von Carlos ermöglicht ihm, Auflistungs- und Leseaktionen für `Resource Y` auszuführen. Die ressourcenbasierte Richtlinie von `Resource Y` gewährt ihm Schreibberechtigungen. Aber obwohl er über seine identitätsbasierte Richtlinie Zugriff auf `Resource Z` erhält, wird ihm dieser Zugriff von der ressourcenbasierten Richtlinie `Resource Z` verweigert. Eine explizite Zugriffsverweigerung `Deny` hat Vorrang vor einer Zugriffserlaubnis `Allow` und sein Zugriff auf `Resource Z` wird verweigert. Weitere Informationen finden Sie unter Auswertungslogik für Richtlinien.
- MaryMajor— Mary kann Listen-, Lese- und Schreiboperationen für `Resource X`Resource Y, und ausführen`Resource Z`. Ihre identitätsbasierte Richtlinie gewährt ihr weitere Aktionen für weitere Ressourcen als die ressourcenbasierten Richtlinien, aber keine davon enthält eine Zugriffsverweigerung.
- ZhangWei— Zhang hat vollen Zugriff auf`Resource Z`. Zhang verfügt über keine identitätsbasierten Richtlinien, erhält jedoch über die ressourcenbasierte Richtlinie von `Resource Z` Vollzugriff auf die Ressource. Zhang kann auch Auflistungs- und Leseaktionen für `Resource Y` ausführen.

Sowohl identitätsbasierte als auch ressourcenbasierte Richtlinien sind Berechtigungsrichtlinien, die gemeinsam ausgewertet werden. Bei einer Anfrage, für die nur Berechtigungsrichtlinien gelten, werden AWS zunächst alle Richtlinien auf eine `Deny` überprüft. Ist eine Zugriffsverweigerung vorhanden, wird die Anfrage abgelehnt. Danach sucht AWS nach jedem `Allow`. Wenn die Aktion in der Anforderung von mindestens einer Richtlinienanweisung gewährt wird, wird die Anforderung gewährt. Dabei spielt es keine Rolle, ob `Allow` in der identitätsbasierten oder der ressourcenbasierten Richtlinie vorhanden ist.

> **Important**
>
> Diese Logik gilt nur, wenn die Anforderung von einem einzelnen AWS-Konto gesendet wird. Bei Anfragen, die von einem Konto an ein anderes gesendet werden, muss der Anforderer in `Account A` über eine identitätsbasierte Richtlinie verfügen, die es ihm ermöglicht, Anforderungen an die Ressource in `Account B` zu senden. Darüber hinaus muss die ressourcenbasierte Richtlinie in `Account B` dem Anforderer in `Account A` Zugriff auf die Ressource gewähren. In beiden Konten müssen Richtlinien vorhanden sein, die die Operation. Andernfalls schlägt die Anforderung fehl. Weitere Informationen zur

Verwendung von ressourcenbasierten Richtlinien für kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.

Ein Benutzer mit speziellen Berechtigungen kann eine Ressource anfordern, an die ebenfalls eine Berechtigungsrichtlinie angefügt ist. In diesem Fall werden bei der Entscheidung, ob Zugriff auf die Ressource gewährt werden soll, beide AWS Berechtigungssätze ausgewertet. Weitere Informationen über das Auswerten von Richtlinien finden Sie unter Auswertungslogik für Richtlinien.

> **Note**
>
> Amazon S3 unterstützt identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien (als Bucket-Richtlinien bezeichnet). Darüber hinaus unterstützt Amazon S3 einen Berechtigungsmechanismus, der auch als Access Control List (ACL) bezeichnet wird, der von IAM-Richtlinien und -Berechtigungen unabhängig ist. Sie können Sie IAM-Richtlinien in Kombination mit Amazon S3-ACLs verwenden. Weitere Informationen finden Sie unter Access Control im Benutzerhandbuch für Amazon Simple Storage Service.

Steuern des Zugriffs auf AWS Ressourcen mithilfe von Richtlinien

Sie können eine Richtlinie verwenden, um den Zugriff auf Ressourcen innerhalb von IAM oder allen Ressourcen zu steuern. AWS

Um eine Richtlinie zur Zugriffskontrolle verwenden zu können AWS, müssen Sie wissen, wie der Zugriff AWS gewährt wird. AWS besteht aus Sammlungen von Ressourcen. Ein IAM-Benutzer ist eine Ressource. Ein Amazon S3-Bucket ist eine Ressource. Wenn Sie die AWS API, die oder die verwenden AWS CLI, AWS Management Console um einen Vorgang auszuführen (z. B. einen Benutzer zu erstellen), senden Sie eine Anfrage für diesen Vorgang. Ihre Anfrage gibt eine Aktion, eine Ressource, eine Prinzipal-Entität (Gruppe oder Rolle) und ein Prinzipalkonto an und enthält alle erforderlichen Anfrageinformationen. Diese Informationen stellen den Kontext bereit.

AWS überprüft dann, ob Sie (der Principal) authentifiziert (angemeldet) und autorisiert (berechtigt) sind, die angegebene Aktion auf der angegebenen Ressource auszuführen. AWS Überprüft während der Autorisierung alle Richtlinien, die für den Kontext Ihrer Anfrage gelten. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert und spezifizieren die Berechtigungen für Hauptentitäten. Weitere Informationen zu diesen Richtlinienarten und ihrer Verwendung finden Sie unter Berechtigungen und Richtlinien in IAM.

AWS autorisiert die Anfrage nur, wenn jeder Teil Ihrer Anfrage gemäß den Richtlinien zulässig ist. Eine Abbildung dieses Prozesses finden Sie unter Funktionsweise von IAM. Einzelheiten dazu, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, finden Sie unterAuswertungslogik für Richtlinien.

Wenn Sie eine IAM-Richtlinie erstellen, können Sie den Zugriff auf Folgendes steuern:

- Prinzipal – Legen Sie fest, welche Aktionen die Person, die die Anfrage stellt (der Prinzipal), durchführen darf.
- IAM-Identitäten – Legen Sie fest, auf welche IAM-Identitäten (Gruppen, Benutzer und Rollen) zugegriffen werden kann und wie.
- IAM-Richtlinien – Legen Sie fest, wer vom Benutzer verwaltete Richtlinien erstellen, bearbeiten und löschen und wer verwaltete Richtlinien anfügen und entfernen kann.
- AWS -Ressourcen – Legen sie fest, wer über eine identitätsbasierte oder ressourcenbasierte Richtlinie Zugriff auf Ressourcen hat.
- AWS -Konten – Legen Sie fest, ob eine Anfrage nur für Mitglieder eines bestimmten Kontos zulässig ist.

Mithilfe von Richtlinien können Sie festlegen, wer Zugriff auf AWS Ressourcen hat und welche Aktionen sie mit diesen Ressourcen ausführen können. Jeder IAM-Benutzer beginnt ohne Berechtigungen. Mit anderen Worten: Benutzer können standardmäßig nichts tun, nicht einmal ihre eigenen Zugriffsschlüssel anzeigen. Um einem Benutzer eine Berechtigung für eine Aktion zu erteilen, können Sie diese zum Benutzer hinzufügen (Sie ordnen dem Benutzer eine Richtlinie zu). Alternativ können Sie den Benutzer einer Gruppe hinzufügen, der die gewünschte Berechtigung bereits zugewiesen wurde.

Sie können beispielsweise einem Benutzer die Berechtigung gewähren, die eigenen Zugriffsschlüssel aufzulisten. Sie können diese Berechtigung auch erweitern, sodass jeder Benutzer die eigenen Schlüssel auch erstellen, aktualisieren und löschen kann.

Wenn Sie einer Gruppe Berechtigungen geben, erhalten alle Benutzer in dieser Gruppe diese Berechtigungen. Beispielsweise können Sie der Benutzergruppe Administratoren die Berechtigung erteilen, alle IAM-Aktionen auf allen AWS-Konto Ressourcen durchzuführen. Ein weiteres Beispiel: Sie können der Gruppe Managers die Berechtigung zum Beschreiben der Amazon-EC2-Instances des AWS-Konto geben.

Weitere Informationen zum Delegieren grundlegender Berechtigungen für Ihre Benutzer, Gruppen und Rollen finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

Weitere Beispiele für Richtlinien, die grundlegende Berechtigungen veranschaulichen, finden Sie unter Beispielrichtlinien für die Verwaltung von IAM-Ressourcen.

Zugriffssteuerung für Prinzipale

Sie können mit Richtlinien steuern, welche Aktionen die Person (der Prinzipal), von der die Anfrage stammt, ausführen darf. Dazu müssen Sie eine identitätsbasierte Richtlinie an die Identität dieser Person (Benutzer, Gruppe von Benutzern oder Rolle) anfügen. Sie können außerdem eine Berechtigungsgrenze zum Festlegen der maximalen Berechtigungen, die eine Entität (Benutzer oder Rolle) haben kann, verwenden.

Nehmen Sie beispielsweise an, dass der Benutzer Zhang Wei vollen Zugriff auf Amazon DynamoDB CloudWatch, Amazon EC2 und Amazon S3 haben soll. In diesem Fall erstellen Sie zwei verschiedene Richtlinien, die Sie später aufbrechen können, wenn Sie einen Berechtigungssatz für einen anderen Benutzer benötigen. Alternativ können Sie beide Berechtigungen in einer einzigen Richtlinie vereinen und diese Richtlinie anschließend dem IAM-Benutzer namens Zhang Wei zuweisen. Sie können auch eine Richtlinie zu einer Gruppe zuweisen, der Zhang angehört, oder einer Rolle zuordnen, die Zhang annehmen kann. Wenn Zhang dann die Inhalte eines S3-Buckets anzeigt, werden die Anfragen zugelassen. Wenn der Benutzer versucht, einen neuen IAM-Bucket zu erstellen, wird die Anfrage aufgrund fehlender Berechtigung abgelehnt.

Sie können eine Berechtigungsgrenze für Zhang verwenden, um sicherzustellen, dass er nie Zugriff auf den S3-Bucket `DOC-EXAMPLE-BUCKET1` erhält. Dazu bestimmen Sie die maximalen Berechtigungen, die Zhang erhalten soll. In diesem Fall steuern Sie, was er unter Verwendung seiner Berechtigungsrichtlinien machen kann. Hier kümmert es Sie nur, dass er nicht auf den vertraulichen Bucket zugreift. Sie verwenden also die folgende Richtlinie, um Zhangs Grenze so zu definieren, dass alle AWS Aktionen für Amazon S3 und einige andere Dienste zulässig sind, aber der Zugriff auf den `DOC-EXAMPLE-BUCKET1` S3-Bucket verweigert wird. Da die Berechtigungsgrenze keine IAM-Aktionen zulässt, verhindert sie, dass Zhang seine Grenze (oder die eines anderen Benutzers) löscht.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
```

```
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ]
        }
    ]
}
```

Wenn Sie eine solche Richtlinie als Berechtigungsgrenze für einen Benutzer zuweisen, denken Sie daran, dass sie keine Berechtigungen gewährt. Sie legt die maximalen Berechtigungen fest, die eine identitätsbasierte Richtlinie einer IAM-Entität erteilen kann. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten.

Weitere Informationen zu den oben genannten Verfahren finden Sie in diesen Ressourcen:

- Weitere Informationen zum Erstellen einer IAM-Richtlinie, die Sie zu einem Prinzipal zuordnen können, finden Sie unter Erstellen von IAM-Richtlinien.
- Weitere Informationen zum Hinzufügen einer IAM-Richtlinie zu einem Prinzipal finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.
- Ein Beispielrichtlinie zum Gewähren eines Vollzugriffs auf EC2 finden Sie unter Amazon EC2: Gewährt EC2-Vollzugriff innerhalb einer bestimmten Region programmatisch und in der Konsole.
- Zum Gewähren eines Lesezugriffs auf einen S3-Bucket verwenden Sie die ersten beiden Anweisungen der folgenden Beispielrichtlinie: Amazon S3: Gewährt Lese- und Schreibzugriff auf Objekte in einem S3-Bucket programmgesteuert und in der Konsole.
- Eine Beispielrichtlinie, mit der Benutzer ihre Anmeldeinformationen wie ihr Konsolenkennwort, ihre programmgesteuerten Zugriffsschlüssel und ihre MFA-Geräte festlegen können, finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Steuern des Zugriffs auf Identitäten

Sie können IAM-Richtlinien verwenden, um zu steuern, was Ihre Benutzer mit einer Identität machen können, indem Sie eine Richtlinie erstellen, die Sie über eine Gruppe allen Benutzern hinzufügen. Dazu erstellen Sie eine Richtlinie, aus denen die Aktionen hervorgehen, die für eine Identität durchgeführt werden dürfen, oder die festlegt, wer zugriffsberechtigt ist.

Sie können beispielsweise eine Benutzergruppe mit dem Namen AllUserserstellen und diese Benutzergruppe dann allen Benutzern zuordnen. Beim Erstellen der Benutzergruppe können Sie allen Benutzern Zugriff darauf gewähren, ihre Anmeldeinformationen wie im vorherigen Abschnitt beschrieben festzulegen. Anschließend können Sie eine Richtlinie erstellen, die einen Zugriff zum Ändern der Gruppe verweigert, sofern der Benutzername nicht Bestandteil der Richtlinienbedingung ist. Aber dieser Teil der Richtlinie verweigert nur jenen Benutzern den Zugriff, die nicht aufgelistet sind. Sie müssen Berechtigungen einschließen, damit alle Gruppenmitglieder Gruppenverwaltungsaktionen durchführen können. Anschließend weisen Sie diese Richtlinie der Gruppe zu, sodass sie für alle Benutzer gilt. Wenn dann ein Benutzer, der nicht in der Richtlinie angegeben ist, versucht, die Gruppe zu ändern, wird die Anfrage abgelehnt.

So erstellen Sie diese Richtlinie mit dem visuellen Editor

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 Create Policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option Visual aus.
5. Wählen Sie unter Select a service (Einen Service auswählen) die Option IAM aus.
6. Geben Sie im Feld Actions allowed (Zulässige Aktionen) **`group`** in das Suchfeld ein. Der visuelle Editor zeigt alle IAM-Aktionen, die das Wort `group` enthalten. Aktivieren Sie alle Kontrollkästchen.
7. Wählen Sie Resources (Ressourcen) aus, um die Ressourcen für Ihre Richtlinie festzulegen. Basierend auf den ausgewählten Aktionen sollten Sie die Ressourcentypen group (Gruppe) und user (Benutzer) sehen.

- group (Gruppe) – Wählen Sie Add ARNs (ARNs hinzufügen) aus. Wählen Sie für Resource in (Ressource in) die Option Any account (Beliebiges Konto) aus. Aktivieren Sie das Kontrollkästchen Any group name with path (Beliebiger Gruppenname mit Pfad) und geben Sie dann den Namen der Benutzergruppe **`AllUsers`** ein. Wählen Sie dann Add ARNs (ARNs hinzufügen) aus.
- user (Benutzer) – Aktivieren Sie das Kontrollkästchen neben Any in this account (Alle in diesem Konto).

Eine der Aktionen, die Sie ausgewählt haben, `ListGroups`, unterstützt die Verwendung spezifischer Ressourcen nicht. Sie müssen für diese Aktion nicht All resources (Alle Ressourcen) auswählen. Beim Speichern oder Anzeigen der Richtlinie im JSON-Editor sehen Sie, dass IAM automatisch einen neuen Berechtigungsblock erstellt, durch den diese Aktion für alle Ressourcen zugelassen wird.

8. Zum Hinzufügen eines weiteren Berechtigungsblocks wählen Sie Add more permissions (Weitere Berechtigungen hinzufügen) aus.
9. Wählen Sie Select a service (Einen Service auswählen) und dann IAM aus.
10. Wählen Sie Actions allowed (Zulässige Aktionen) und dann Switch to deny permissions (Zu verweigerten Berechtigungen wechseln) aus. Wenn Sie dies durchführen, wird der gesamte Block verwendet, um Berechtigungen zu verweigern.
11. Geben Sie in das Suchfeld **`group`** ein. Die visuelle Editor zeigt alle IAM-Aktionen an, die das Wort `group` enthalten. Aktivieren Sie die Kontrollkästchen neben den folgenden Aktionen:

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy
    - DetachGroupPolicy
    - PutGroupPolicy
    - UpdateGroup
12. Wählen Sie Resources (Ressourcen) aus, um die Ressourcen für Ihre Richtlinie festzulegen. Basierend auf den Aktionen, die Sie ausgewählt haben, sollten Sie den Ressourcentyp group (Gruppe) sehen. Wählen Sie Add ARNs (ARNs hinzufügen) aus. Wählen Sie für Resource in

(Ressource in) die Option Any account (Beliebiges Konto) aus. Geben Sie für any group Name With Path (Jeder Gruppenname mit Pfad) den Gruppennamen **`AllUsers`** ein. Wählen Sie dann Add ARNs (ARNs hinzufügen) aus.

13. Wählen Sie Request conditions - optional (Anforderungsbedingungen - optional) und anschließend Add another condition (Weitere Bedingung hinzufügen). Füllen Sie das Formular mit den folgenden Werten aus:

    - Condition key (Bedingungsschlüssel) – Wählen Sie aws:username aus.
    - Qualifier (Qualifizierer) – Wählen Sie Default (Standard)
    - Betreiber — Wählen Sie StringNotEquals
    - Value (Wert) – Geben Sie **`srodriguez`** ein und wählen Sie dann Add (Hinzufügen) aus, um einen weiteren Wert hinzuzufügen. Geben Sie **`mjackson`** ein und wählen Sie dann Add (Hinzufügen) aus, um einen weiteren Wert hinzuzufügen. Geben Sie **`adesai`** ein und wählen Sie Add condition (Bedingung hinzufügen).

    Diese Bedingung stellt sicher, dass der Zugriff auf die angegebenen Gruppenverwaltungsaktionen verweigert wird, wenn der Benutzer, der den Aufruf durchführt, nicht in der Liste enthalten ist. Da dadurch die Berechtigung explizit verweigert wird, wird der vorherige Block überschrieben, der den Benutzern das Aufrufen der Aktionen gestattete. Benutzern auf der Liste wird der Zugriff nicht verweigert. Sie erhalten eine Berechtigung im ersten Block, sodass sie die Gruppe vollständig verwalten können.

14. Wählen Sie danach Next aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder in der Editoroption Visual Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

15. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) für Policy Name (Richtlinienname) **`LimitAllUserGroupManagement`** ein. Geben Sie für Description (Beschreibung) Folgendes ein: **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um sicherzustellen, dass Sie die beabsichtigten Berechtigungen

erteilt haben. Wählen Sie dann Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

16. Fügen Sie die Richtlinie zu Ihrer Gruppe hinzu. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

Alternativ können Sie dieselbe Richtlinie mit diesem Beispieldokument einer JSON-Richtlinie erstellen. So zeigen Sie diese JSON-Richtlinie finden Sie unter : Ermöglicht spezifischen IAM-Benutzern das Verwalten einer Gruppe programmgesteuert und in der Konsole. Detaillierte Anweisungen zum Erstellen einer Richtlinie mithilfe eines JSON-Dokuments finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

Steuern des Zugriffs auf Richtlinien

Sie können steuern, wie Ihre Benutzer AWS verwaltete Richtlinien anwenden können. Zu diesem Zweck weisen Sie diese Richtlinie all Ihren Benutzern zu. Idealerweise können Sie dies durch die Verwendung einer Gruppe durchführen.

Sie können beispielsweise eine Richtlinie erstellen, die es Benutzern ermöglicht, einem neuen IAM-Benutzer, einer neuen Benutzergruppe oder einer neuen IAM-Rolle nur das IAM UserChangePassword und die PowerUserAccess AWS verwalteten Richtlinien zuzuordnen.

Für vom Kunden verwaltete Richtlinien können Sie steuern, wer diese Richtlinien erstellen, aktualisieren und löschen darf. Sie können steuern, welche Benutzer Richtlinien zu Prinzipal-Entitäten (Gruppen, Benutzer und Rollen) zuordnen und von diesen entfernen können. Sie können auch steuern, welche Richtlinien ein Benutzer welchen Entitäten hinzufügen oder von diesen trennen kann.

So können Sie beispielsweise einem Kontoadministrator Berechtigungen zum Erstellen, Aktualisieren und Löschen von Richtlinien erteilen. Anschließend erteilen Sie einem Teamleiter oder einem anderen Administrator mit eingeschränkten Befugnissen die Berechtigungen, um diese Richtlinien den vom Administrator mit eingeschränkten Befugnissen verwalteten Prinzipal-Entitäten anzufügen und von diesen zu trennen.

Weitere Informationen finden in folgenden Ressourcen:

- Weitere Informationen zum Erstellen einer IAM-Richtlinie, die Sie zu einem Prinzipal zuordnen können, finden Sie unter Erstellen von IAM-Richtlinien.
- Weitere Informationen zum Hinzufügen einer IAM-Richtlinie zu einem Prinzipal finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

- Ein Beispiel für eine Richtlinie, die die Verwendung verwalteter Richtlinien einschränkt, finden Sie unter IAM: Beschränkt die verwalteten Richtlinien, die -Benutzern, -Gruppen oder -Rollen zugeordnet werden können.

Kontrollieren der Berechtigungen zum Erstellen, Aktualisieren und Löschen von Berechtigungen für vom Kunden verwaltete Berechtigungen

Sie können mit IAM-Richtlinien steuern, wer vom Kunden verwaltete Richtlinien in Ihrem AWS-Konto erstellen, aktualisieren und löschen darf. Die folgende Liste enthält API-Operationen, die speziell für das Erstellen, Aktualisieren und Löschen von Richtlinien oder Richtlinienversionen vorgesehen sind:

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

Die API-Operationen in der vorherigen Liste entsprechen Aktionen, die Sie zulassen oder verweigern können, das heißt, Berechtigungen, die Sie mit mithilfe einer IAM-Richtlinie erteilen können.

Betrachten Sie die folgende Beispielrichtlinie. Sie erteilt einem Benutzer die Berechtigung zum Erstellen, Aktualisieren (d. h. zum Erstellen einer neuen Richtlinienversion), Löschen und Festlegen einer Standardversion für alle vom Kunden verwalteten Richtlinien im AWS-Konto. Die Beispielrichtlinie gestattet dem Benutzer auch das Auflisten und Abrufen von Richtlinien. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

Example Beispielrichtlinie, die das Erstellen, Aktualisieren, Löschen, Auflisten, Abrufen und Einstellen der Standardversion für alle Richtlinien ermöglicht

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
```

```
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

Sie können Richtlinien erstellen, die die Verwendung dieser API-Operationen auf die von Ihnen angegebenen, verwalteten Richtlinien beschränken. Sie möchten beispielsweise einem Benutzer die Berechtigung zum Festlegen der Standardversion und Löschen von Richtlinienversionen ausschließlich für bestimmte, vom Kunden verwaltete Richtlinien erteilen. Geben Sie hierzu den ARN der Richtlinie im Element `Resource` der Richtlinie an, die diese Berechtigungen erteilt

Das folgende Beispiel zeigt eine Richtlinie, mit der ein Benutzer Richtlinienversionen löschen und die Standardversion festlegen kann. Diese Aktionen sind jedoch nur für die vom Kunden verwalteten Richtlinien erlaubt, die den Pfad /TEAM-A/ enthalten. Der ARN der kundenverwalteten Richtlinie ist im Element `Resource` der Richtlinie angegeben. (In diesem Beispiel enthält der ARN einen Pfad und ein Platzhalterzeichen und stimmt so mit allen kundenverwalteten Richtlinien überein, die den Pfad /TEAM-A/ enthalten). Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

Weitere Informationen zur Verwendung von Pfaden in den Namen der von Kunden verwalteten Richtlinien finden Sie unter Anzeigenamen und -pfade.

Example Beispielrichtlinie, die das Löschen von Richtlinienversionen und das Einstellen der Standardversion nur für bestimmte Richtlinien erlaubt

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:DeletePolicyVersion",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "arn:aws:iam::account-id:policy/TEAM-A/*"
```

```
  }
}
```

Steuern der Berechtigungen für das Anfügen und Trennen von verwalteten Richtlinien

Sie können auch IAM-Richtlinien verwenden, damit Benutzer nur mit bestimmten verwalteten Richtlinien arbeiten können. Sie können sogar steuern, welche Berechtigungen ein Benutzer anderen Prinzipal-Entitäten erteilen kann.

In der folgenden Liste werden API-Operationen aufgeführt, die sich speziell auf das Anfügen und Trennen von verwalteten Richtlinien an und von Prinzipal-Entitäten beziehen:

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Sie können Richtlinien erstellen, die die Verwendung dieser API-Operationen auf die von Ihnen angegebenen, verwalteten Richtlinien und/oder Prinzipal-Entitäten beschränken. Sie möchten beispielsweise einem Benutzer die Berechtigung erteilen, ausschließlich die von Ihnen angegebenen, verwalteten Richtlinien anzufügen. Oder Sie möchten einem Benutzer die Berechtigung erteilen, die verwalteten Richtlinien ausschließlich den von Ihnen angegebenen Prinzipal-Entitäten anzufügen.

Die folgende Beispielrichtlinie erteilt einem Benutzer die Berechtigung, verwaltete Richtlinien ausschließlich den Gruppen und Rollen anzufügen, die den Pfad /TEAM-A/ enthalten. Die Gruppen- und Rollen-ARNs sind im `Resource`-Element der Richtlinie angegeben. (In diesem Beispiel enthalten die ARNs einen Pfad und ein Platzhalterzeichen und stimmen so mit allen Gruppen und Rollen überein, die den Pfad /TEAM-A / enthalten). Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

Example Beispielrichtlinie, die es ermöglicht, verwaltete Richtlinien nur bestimmten Benutzergruppen oder Rollen zuzuordnen

```
{
```

```
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ]
  }
}
```

Sie können außerdem die Aktionen im vorherigen Beispiel auf bestimmte Richtlinien einschränken. Das heißt, Sie können steuern, welche Berechtigungen ein Benutzer anderen Prinzipal-Entitäten anfügen kann indem Sie eine Bedingung der Richtlinie hinzufügen.

Im folgenden Beispiel stellt die Bedingung sicher, dass die Berechtigungen `AttachGroupPolicy` und `AttachRolePolicy` nur dann zulässig sind, wenn die angefügte Richtlinie mit einer der angegebenen Richtlinien übereinstimmt. Die Bedingung verwendet den `iam:PolicyARN` Bedingungsschlüssel, um festzulegen, welche Richtlinie oder Richtlinien angefügt werden dürfen. Die folgende Beispielrichtlinie erweitert das vorherige Beispiel. Es erlaubt einem Benutzer, nur die verwalteten Richtlinien, die den Pfad /TEAM-A/ enthalten, den Gruppen und Rollen hinzuzufügen, die den Pfad /TEAM-A/ enthalten. Informationen zum Erstellen einer Richtlinie mit diesem Beispiel-JSON-Richtliniendokument finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ],
    "Condition": {"ArnLike":
      {"iam:PolicyARN": "arn:aws:iam::account-id:policy/TEAM-A/*"}
```

```
    }
  }
}
```

Diese Richtlinie verwendet den `ArnLike`-Bedingungsoperator, aber Sie können auch den `ArnEquals`-Bedingungsoperator verwenden, weil sich diese beiden Bedingungsoperatoren identisch verhalten. Weitere Informationen zu `ArnLike` und `ArnEquals` finden Sie unter Bedingungsoperatoren für Amazon-Ressourcennamen (ARN) im Abschnitt Bedingungstypen in der Richtinienelementreferenz.

Sie können beispielsweise die Verwendung dieser Aktionen einschränken, sodass nur die von Ihnen angegebenen, verwalteten Richtlinien einbezogen werden. Geben Sie hierzu den ARN der Richtlinie im Element `Condition` der Richtlinie an, die diese Berechtigungen erteilt Wenn Sie beispielsweise den ARN einer vom Kunden verwalteten Richtlinie angeben möchten:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

Sie können auch den ARN einer AWS verwalteten Richtlinie im `Condition` Element einer Richtlinie angeben. Der ARN einer AWS verwalteten Richtlinie verwendet den speziellen Alias `aws` im Richtlinien-ARN anstelle einer Konto-ID, wie in diesem Beispiel:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Steuern des Zugriffs auf Ressourcen

Sie können den Zugriff auf Ressourcen mit einer identitätsbasierten oder ressourcenbasierten Richtlinie steuern. Bei einer identitätsbasierten Richtlinie fügen Sie die Richtlinie einer Identität hinzu und geben an, auf welche Ressourcen die Identität zugreifen darf. Bei einer ressourcenbasierten Richtlinie ordnen Sie eine Richtlinie der Ressource zu, die Sie steuern möchten. Sie geben in der Richtlinie an, welche Prinzipale auf die Ressource zugreifen dürfen. Weitere Informationen zu diesen beiden Arten von Richtlinien finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien.

Weitere Informationen finden in folgenden Ressourcen:

- Weitere Informationen zum Erstellen einer IAM-Richtlinie, die Sie zu einem Prinzipal zuordnen können, finden Sie unter Erstellen von IAM-Richtlinien.
- Weitere Informationen zum Hinzufügen einer IAM-Richtlinie zu einem Prinzipal finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.
- Amazon S3 unterstützt die Verwendung von ressourcenbasierten Richtlinien für die Buckets. Weitere Informationen finden Sie unter Bucket Policy Examples.

Ersteller von Ressourcen haben nicht automatisch Berechtigungen

Wenn Sie sich mit den Root-Benutzer des AWS-Kontos Anmeldeinformationen anmelden, sind Sie berechtigt, alle Aktionen mit Ressourcen durchzuführen, die zu dem Konto gehören. Dies gilt jedoch nicht für IAM-Benutzer. Einem IAM-Benutzer kann der Zugriff zum Erstellen einer Ressource gewährt werden, aber die Berechtigungen des Benutzers, selbst für diese Ressource, sind darauf beschränkt, was explizit erteilt wurde. Das bedeutet: Nur weil Sie eine Ressource, beispielsweise eine IAM-Rolle, erstellen, sind Sie nicht automatisch berechtigt, diese Rolle zu bearbeiten oder zu löschen. Darüber hinaus kann Ihre Berechtigung jederzeit vom Kontoinhaber oder einem anderen Benutzer widerrufen werden, dem der Zugriff auf die Verwaltung Ihrer Berechtigungen gewährt wurde.

Steuern des Zugriffs auf Prinzipale in einem bestimmten Konto

Sie können direkt in Ihrem eigenen Konto IAM-Benutzern Zugriff auf Ihre Ressourcen gewähren. Wenn Benutzer eines anderen Kontos Zugriff auf Ihre Ressourcen benötigen, können Sie eine IAM-Rolle erstellen. Eine Rolle ist eine Entität, die Berechtigungen enthält, aber nicht mit einem bestimmten Benutzer verknüpft ist. Benutzer von anderen Konten können dann die Rolle annehmen und entsprechend den Berechtigungen, die Sie der Rolle zugeordnet haben, auf Ressourcen zugreifen. Weitere Informationen finden Sie unter Bereitstellen des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

> **Note**
>
> Einige Services unterstützen ressourcenbasierte Richtlinien wie in Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien beschrieben (wie Amazon S3, Amazon SNS und Amazon SQS). Für diese Services besteht eine Alternative zur Verwendung von Rollen darin, eine Richtlinie an die Ressource (Bucket, Thema oder Warteschlange) anzuhängen, die Sie freigeben möchten. Die ressourcenbasierte Richtlinie kann das AWS Konto angeben, das über Berechtigungen für den Zugriff auf die Ressource verfügt.

Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags

Verwenden Sie die Informationen im folgenden Abschnitt, um zu steuern, wer auf Ihre IAM-Benutzer und IAM-Rollen zugreifen kann und auf welche Ressourcen Ihre Benutzer und Rollen zugreifen können. Allgemeinere Informationen und Beispiele für die Steuerung des Zugriffs auf andere AWS Ressourcen, einschließlich anderer IAM-Ressourcen, finden Sie unterMarkieren von IAM-Ressourcen.

> **Note**
>
> Einzelheiten zur Berücksichtigung der Groß- und Kleinschreibung bei Tag-Schlüsseln und Tag-Schlüsselwerten finden Sie unter Case sensitivity.

Tags können an die IAM-Ressource angehängt, in der Anforderung übergeben oder an den Auftraggeber, der die Anforderung stellt, angehängt werden. Ein Benutzer oder eine Rolle in IAM kann sowohl eine Ressource als auch ein Auftraggeber sein. Sie können beispielsweise eine Richtlinie schreiben, mit der ein Benutzer die Gruppen für einen Benutzer auflisten kann. Diese Operation ist nur zulässig, wenn der Benutzer, der die Anforderung stellt (der Auftraggeber), über das gleiche `project=blue`-Tag verfügt wie der Benutzer, den er anzeigen möchte. In diesem Beispiel kann der Benutzer die Gruppenmitgliedschaft für jeden Benutzer anzeigen, einschließlich sich selbst, solange sie an demselben Projekt arbeiten.

Um den Zugriff auf Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an. Wenn Sie eine IAM-Richtlinie erstellen, können Sie IAM-Tags und den zugehörigen Tag-Bedingungsschlüssel verwenden, um den Zugriff auf Folgendes zu steuern:

- Ressource – Steuern Sie den Zugriff auf Benutzer- oder Rollenressourcen basierend auf ihren Tags. Verwenden Sie dazu den Bedingungsschlüssel aws:ResourceTag/***key-name***, um anzugeben, welches Tag-Schlüssel-Wert-Paar an die Ressource angehängt werden muss. Weitere Informationen finden Sie unter Steuern des Zugriffs auf AWS -Ressourcen.
- Anforderung – Bestimmen, welche Tags in einer IAM-Anforderung weitergeleitet werden können. Verwenden Sie dazu den Bedingungsschlüssel aws:RequestTag/für ***Schlüsselnamen***, um anzugeben, welche Tags einem IAM-Benutzer oder einer IAM-Rolle hinzugefügt, geändert oder daraus entfernt werden können. Dieser Schlüssel wird auf die gleiche Weise für IAM-Ressourcen

und andere Ressourcen verwendet. AWS Weitere Informationen finden Sie unter Zugriffssteuerung während AWS -Anforderungen.

- Auftraggeber – Steuern Sie, welche Aktionen die Person, von der die Anforderung stammt (der Auftraggeber), durchführen darf, auf Grundlage der Tags, die dem IAM-Benutzer oder der Rolle der Person angefügt sind. Verwenden Sie dazu den Bedingungsschlüssel aws:PrincipalTag/***key-name***, um anzugeben, welche Tags an den IAM-Benutzer oder die IAM-Rolle angehängt werden müssen, bevor die Anfrage zugelassen wird.
- Beliebiger Teil des Autorisierungsprozesses — Verwenden Sie den TagKeys Bedingungsschlüssel aws:, um zu steuern, ob bestimmte Tag-Schlüssel in einer Anfrage oder von einem Principal verwendet werden können. In diesem Fall spielt der Schlüsselwert keine Rolle. Dieser Schlüssel verhält sich bei IAM und anderen AWS Diensten ähnlich. Wenn Sie jedoch einen Benutzer in IAM markieren, steuert dies auch, ob der Auftraggeber die Anforderung an einen beliebigen Service stellen kann. Weitere Informationen finden Sie unter Zugriffssteuerung auf der Grundlage von Tag-Schlüsseln.

Sie können eine IAM-Richtlinie mit dem visuellen Editor, JSON oder durch Importieren einer vorhandenen verwalteten Richtlinie erstellen. Details hierzu finden Sie unter Erstellen von IAM-Richtlinien.

> **Note**
>
> Sie können Sitzungs-Tags auch übergeben, wenn Sie eine IAM-Rolle übernehmen oder einen Benutzer in einen Verbund aufnehmen. Diese sind nur für die Dauer der Sitzung gültig.

Zugriffssteuerung für IAM-Auftraggeber

Sie können steuern, was der Auftraggeber tun darf, basierend auf den Tags, die der Identität dieser Person zugeordnet sind.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es jedem Benutzer in diesem Konto erlaubt, die Gruppenmitgliedschaft für jeden Benutzer, einschließlich sich selbst, anzuzeigen, solange sie am selben Projekt arbeiten. Diese Operation ist nur zulässig, wenn das Ressourcen-Tag des Benutzers und das Tag des Auftraggebers denselben Wert für den Tag-Schlüssel besitzen. `project`. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
        }]
}
```

Zugriffssteuerung auf der Grundlage von Tag-Schlüsseln

Sie können Tags in Ihren IAM-Richtlinien verwenden, um zu steuern, ob bestimmte Tag-Schlüssel in einer Anfrage oder von einem Principal verwendet werden können.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es erlaubt, nur das Tag mit dem Schlüssel `temporary` von Benutzern zu entfernen. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": ["temporary"]}}
    }]
}
```

Steuern des Zugriffs auf AWS Ressourcen mithilfe von Tags

Sie können Tags verwenden, um den Zugriff auf Ihre AWS Ressourcen zu steuern, die Tagging unterstützen, einschließlich IAM-Ressourcen. Sie können IAM-Benutzer und IAM-Rollen markieren, um zu steuern, worauf sie zugreifen können. Weitere Informationen zum Markieren von IAM-

Benutzern und IAM-Rollen finden Sie unter Markieren von IAM-Ressourcen. Darüber hinaus können Sie den Zugriff auf die folgenden IAM-Ressourcen steuern: von Kunden verwaltete Richtlinien, IAM-Identitätsanbieter, Instance-Profile, Serverzertifikate und virtuelle MFA-Geräte. Ein Tutorial zum Erstellen und Testen einer Richtlinie, die IAM-Rollen mit Auftraggeber-Tags den Zugriff auf Ressourcen mit übereinstimmenden Tags erlaubt, finden Sie unter IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren. Verwenden Sie die Informationen im folgenden Abschnitt, um den Zugriff auf andere AWS Ressourcen, einschließlich IAM-Ressourcen, zu steuern, ohne IAM-Benutzer oder -Rollen zu taggen.

Bevor Sie Tags verwenden, um den Zugriff auf Ihre AWS Ressourcen zu steuern, müssen Sie wissen, wie AWS der Zugriff gewährt wird. AWS besteht aus Sammlungen von Ressourcen. Eine Amazon EC2-Instance ist eine Ressource. Ein Amazon S3-Bucket ist eine Ressource. Sie können die AWS API, die oder die verwenden AWS CLI, AWS Management Console um einen Vorgang auszuführen, z. B. das Erstellen eines Buckets in Amazon S3. Wenn Sie dies tun, senden Sie eine Anforderung für diese Operation. Ihre Anforderung gibt eine Aktion, eine Ressource, eine Auftraggeber-Entität (Gruppe oder Rolle) und ein Auftraggeberkonto an und enthält alle erforderlichen Anforderungsinformationen. Diese Informationen stellen den Kontext bereit.

AWS überprüft dann, ob Sie (die Haupteinheit) authentifiziert (angemeldet) und autorisiert (berechtigt) sind, die angegebene Aktion auf der angegebenen Ressource auszuführen. AWS Überprüft während der Autorisierung alle Richtlinien, die für den Kontext Ihrer Anfrage gelten. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert und spezifizieren die Berechtigungen für Hauptentitäten. Weitere Informationen zu diesen Richtlinienarten und ihrer Verwendung finden Sie unter Berechtigungen und Richtlinien in IAM.

AWS autorisiert die Anfrage nur, wenn jeder Teil Ihrer Anfrage gemäß den Richtlinien zulässig ist. Informationen zum Anzeigen eines Diagramms und weitere Informationen über die IAM-Infrastruktur finden Sie unter Funktionsweise von IAM. Weitere Informationen darüber, wie IAM bestimmt, ob eine Anfoderung zulässig ist, finden Sie unter Auswertungslogik für Richtlinien.

Tags sind eine weitere Überlegung in diesem Prozess, da Markierungen an die Ressource angefügt werden können oder in der Anfrage an Services weitergegeben werden können, die das Markieren unterstützen. Um den Zugriff auf Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an. Informationen darüber, ob ein AWS Service die Zugriffskontrolle mithilfe von Tags unterstützt, finden Sie unter AWS Dienste, die mit IAM funktionieren und suchen Sie nach den Diensten, für die in der ABAC-Spalte Ja angegeben ist. Wählen Sie den Namen des Service, um die Dokumentation zur Autorisierung und Zugriffssteuerung für diesen Service anzuzeigen.

Sie können dann eine IAM-Richtlinie erstellen, die den Zugriff auf eine Ressource basierend auf dem Tag dieser Ressource erlaubt oder verweigert. In dieser Richtlinie können Sie Tag-Bedingungsschlüssel verwenden, um den Zugriff auf eine der folgenden Optionen zu steuern:

- Ressource — Steuern Sie den Zugriff auf AWS Serviceressourcen anhand der Tags auf diesen Ressourcen. Verwenden Sie dazu den Bedingungsschlüssel ResourceTag/***key-name***, um anhand der Tags, die der Ressource zugeordnet sind, zu bestimmen, ob der Zugriff auf die Ressource erlaubt werden soll.
- Anforderung – Steuern, welche Tags an eine Anforderung weitergeleitet werden können. Verwenden Sie dazu den Bedingungsschlüssel aws:RequestTag/***Key-Name***, um anzugeben, welche Tag-Schlüssel-Wert-Paare in einer Anfrage zur Kennzeichnung einer Ressource übergeben werden können. AWS
- Beliebiger Teil des Autorisierungsprozesses — Verwenden Sie den TagKeys Bedingungsschlüssel aws:, um zu kontrollieren, ob bestimmte Tag-Schlüssel in einer Anfrage enthalten sein können.

Sie können eine IAM-Richtlinie visuell mithilfe von JSON oder durch Importieren einer vorhandenen verwalteten Richtlinie erstellen. Details hierzu finden Sie unter Erstellen von IAM-Richtlinien.

> **Note**
>
> Bei einigen Diensten können Benutzer beim Erstellen der Ressource Tags angeben, wenn sie über die Berechtigung zum Verwenden der Aktion verfügen, mit der die Ressource erstellt wird.

Steuern des Zugriffs auf AWS -Ressourcen

Sie können Bedingungen in Ihren IAM-Richtlinien verwenden, um den Zugriff auf AWS Ressourcen anhand der Tags auf dieser Ressource zu steuern. Sie können dies mithilfe des globalen `aws:ResourceTag/`*`tag-key`*-Bedingungsschlüssels oder eines servicespezifischen Schlüssels erreichen. Einige Services unterstützen nur die servicespezifische Version dieses Schlüssels und nicht die globale Version.

> **Warning**
>
> Versuchen Sie nicht zu kontrollieren, wer eine Rolle weitergeben kann, indem Sie die Rolle taggen und dann den `ResourceTag`-Bedingungsschlüssel in einer Richtlinie mit der

`iam:PassRole`-Aktion verwenden. Dieser Ansatz führt nicht zu zuverlässigen Ergebnissen. Weitere Informationen zu den erforderlichen Berechtigungen, die für das Übergeben einer Rolle an einen Service erforderlich sind, finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Starten oder Stoppen von Amazon-EC2-Instances erlaubt. Diese Operationen sind nur zulässig, wenn das Instance-Tag `Owner` den Wert des Benutzernamens dieses Benutzers aufweist. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

Sie können diese Richtlinie den IAM-Benutzern in Ihrem Konto anfügen. Wenn ein Benutzer namens `richard-roe` versucht, eine Amazon EC2-Instance zu starten, muss die Instance mit dem Tag `Owner=richard-roe` oder `owner=richard-roe` versehen sein. Andernfalls wird ihm der Zugriff verweigert. Der Tag-Schlüssel `Owner` stimmt mit `Owner` und `owner` überein, da bei Bedingungsschlüsselnamen nicht zwischen Groß- und Kleinschreibung unterschieden wird. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Condition.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die das Löschen von Benutzern mit dem `team`-Prinzipal-Tag erlaubt. Die Richtlinie gewährt die Erlaubnis, Warteschlangen von Amazon Simple Queue Service zu löschen, jedoch nur, wenn der Warteschlangenname mit dem Teamnamen beginnt, gefolgt von `-queue`. Beispielsweise `qa-queue`, wenn `qa` der Teamname für das `team`-Prinzipal-Tag ist.

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2::${aws:PrincipalTag/team}-queue"
      }
}
```

Zugriffssteuerung während AWS -Anforderungen

Sie können Bedingungen in Ihren IAM-Richtlinien verwenden, um zu steuern, welche Tag-Schlüssel-Wert-Paare in einer Anfrage übergeben werden können, die Tags auf eine Ressource anwendet. AWS

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es erlaubt, mithilfe der Amazon-EC2-Aktion `CreateTags` Tags an eine Instance anzuhängen. Sie können Tags nur anfügen, wenn das Tag den `environment`-Schlüssel und die `preprod`- oder `production`-Werte enthält. Wenn Sie möchten, können Sie den `ForAllValues`-Modifikator mit dem `aws:TagKeys`-Bedingungsschlüssel verwenden, um anzugeben, dass nur der `environment`-Schlüssel in der Anforderung zulässig ist. So wird verhindert, dass Benutzer andere Schlüssel einbeziehen und etwa versehentlich `Environment` anstelle von `environment` verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
```

```
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Zugriffssteuerung auf der Grundlage von Tag-Schlüsseln

Sie können in Ihren IAM-Richtlinien eine Bedingung verwenden, um zu steuern, ob bestimmte Tag-Schlüssel in einer Anfrage verwendet werden können.

Wir empfehlen, dass Sie bei der Verwendung von Richtlinien zur Zugriffskontrolle mithilfe von Tags den `aws:TagKeys` Bedingungsschlüssel verwenden. AWS Dienste, die Tags unterstützen, ermöglichen es Ihnen möglicherweise, mehrere Tag-Schlüsselnamen zu erstellen, die sich nur durch Groß- und Kleinschreibung unterscheiden, z. B. das Taggen einer Amazon EC2 EC2-Instance mit `stack=production` und. `Stack=test` Bei den Schlüsselnamen in den Richtlinienbedingungen wird nicht zwischen Groß- und Kleinschreibung unterschieden. Dies bedeutet Folgendes: Wenn Sie `"aws:ResourceTag/TagKey1": "Value1"` im Bedingungselement Ihrer Richtlinie angeben, stimmt die Bedingung mit einem Ressourcen-Tag-Schlüssel mit dem Namen `TagKey1` oder `tagkey1` überein, aber nicht mit beiden. Um doppelte Tags mit Schlüsseln zu verhindern, die sich nur in der Groß-/Kleinschreibung unterscheiden, verwenden Sie die `aws:TagKeys`-Bedingung zum Definieren von Tag-Schlüsseln, die Ihre Benutzer anwenden können,, oder verwenden Sie Tag-Richtlinien, die mit AWS Organizations verfügbar sind. Weitere Informationen finden Sie unter Tag-Richtlinien im Benutzerhandbuch für Organisationen.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Erstellen und Markieren eines Secrets Manager Geheimnisses erlaubt, jedoch nur mit den Tag-Schlüsseln `environment` oder `cost-center`. Die `Null`-Bedingung stellt sicher, dass die Bedingung zu `false` ausgewertet wird, wenn in der Anfrage keine Tags vorhanden sind.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
```

```
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

Kontoübergreifender Zugriff auf Ressourcen in IAM

Für einige AWS Dienste können Sie mithilfe von IAM kontoübergreifenden Zugriff auf Ihre Ressourcen gewähren. Dazu können Sie eine Ressourcenrichtlinie direkt an die Ressource anfügen, die Sie freigeben möchten, oder eine Rolle als Proxy verwenden.

Um die Ressource direkt freizugeben, muss die Ressource, die Sie freigeben möchten, ressourcenbasierte Richtlinien unterstützen. Im Gegensatz zu einer identitätsbasierten Richtlinie für eine Rolle legt eine ressourcenbasierte Richtlinie fest, wer (welcher Prinzipal) auf diese Ressource zugreifen kann.

Verwenden Sie eine Rolle als Proxy, wenn Sie auf Ressourcen in einem anderen Konto zugreifen möchten, die keine ressourcenbasierten Richtlinien unterstützen.

Einzelheiten zu den Unterschieden zwischen diesen Richtlinientypen finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien.

> **Note**
>
> IAM-Rollen und ressourcenbasierte Richtlinien delegieren den Zugriff auf Konten nur innerhalb einer einzelnen Partition. Beispiel: Sie haben ein Konto in USA West (Nordkalifornien) in der Standardpartition `aws`. Sie haben auch ein Konto in China in der Partition `aws-cn`. Sie können in Ihrem Konto in China keine ressourcenbasierte Richtlinie verwenden, um Benutzern in Ihrem Standardkonto Zugriff zu gewähren. AWS

Kontoübergreifender Zugriff mithilfe von Rollen

Nicht alle AWS Dienste unterstützen ressourcenbasierte Richtlinien. Für diese Services können Sie kontoübergreifende IAM-Rollen verwenden, um die Berechtigungsverwaltung zu zentralisieren,

wenn Sie kontoübergreifenden Zugriff für mehrere Services bereitstellen. Eine kontoübergreifende IAM-Rolle ist eine IAM-Rolle, die eine Vertrauensrichtlinie beinhaltet, die es IAM-Prinzipalen in einem anderen Konto ermöglicht, diese Rolle zu übernehmen. AWS Einfach ausgedrückt können Sie in einem Konto eine Rolle erstellen, die bestimmte Berechtigungen an ein AWS anderes Konto delegiert. AWS

Informationen zum Zuweisen einer Richtlinie an eine IAM-Identität finden Sie unter Verwalten von IAM-Richtlinien.

> **Note**
>
> Wenn ein Prinzipal zu einer Rolle wechselt, um vorübergehend die Berechtigungen der Rolle zu nutzen, gibt er seine ursprünglichen Berechtigungen auf und übernimmt die Berechtigungen, die der Rolle zugewiesen wurden, die er angenommen hat.

Schauen wir uns den gesamten Prozess an, der für die APN-Partnersoftware gilt, die auf ein Kundenkonto zugreifen muss.

1. Der Kunde erstellt in seinem eigenen Konto eine IAM-Rolle mit einer Richtlinie, die den Zugriff auf die Amazon-S3-Ressourcen ermöglicht, die der APN-Partner benötigt. In diesem Beispiel lautet der Rollenname `APNPartner`.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "s3:*",
               "Resource": [
                   "arn:aws:s3:::bucket-name"
               ]
           }
       ]
   }
   ```

2. Anschließend gibt der Kunde an, dass die Rolle vom AWS Konto des Partners übernommen werden kann, indem er die AWS-Konto ID des APN-Partners in der Vertrauensrichtlinie für die `APNPartner` Rolle angibt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::APN-account-ID:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Der Kunde gibt dem APN-Partner den Amazon-Ressourcennamen (ARN) der Rolle. Der ARN ist der vollqualifizierte Name der Rolle.

```
arn:aws:iam::APN-ACCOUNT-ID:role/APNPartner
```

> **Note**
>
> Wir empfehlen, in Situationen mit mehreren Mandanten eine externe ID zu verwenden. Details hierzu finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.

4. Wenn die Software des APN-Partners auf das Konto des Kunden zugreifen muss, ruft die Software die AssumeRoleAPI AWS Security Token Service mit dem ARN der Rolle im Kundenkonto auf. STS gibt einen temporären AWS Berechtigungsnachweis zurück, der es der Software ermöglicht, ihre Arbeit zu erledigen.

Ein weiteres Beispiel für die Gewährung von kontoübergreifendem Zugriff mithilfe von Rollen finden Sie unter Bereitstellen des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen. Sie können auch dem Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen folgen.

Kontoübergreifender Zugriff mit ressourcenbasierten Richtlinien

Wenn ein Konto über ein anderes Konto unter Verwendung einer ressourcenbasierten Richtlinie auf eine Ressource zugreift, funktioniert der Prinzipal weiterhin mit dem vertrauenswürdigen Konto und

muss seine Berechtigungen nicht aufgeben, um die Rollenberechtigungen zu erhalten. Mit anderen Worten, der Prinzipal hat nach wie vor Zugriff auf Ressourcen im vertrauenswürdigen Konto, aber gleichzeitig auch Zugriff auf die Ressource im vertrauenden Konto. Dies ist nützlich für Aufgaben wie das Kopieren von Daten in die oder aus der gemeinsam genutzten Ressource im anderen Konto.

Zu den Prinzipalen, die Sie in einer ressourcenbasierten Richtlinie angeben können, gehören Konten, IAM-Benutzer, Verbundbenutzer, IAM-Rollen, Sitzungen mit übernommenen Rollen oder Dienste. AWS Weitere Informationen finden Sie unter Angeben eines Auftraggebers.

Informationen darüber, ob Prinzipalen in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Identifizieren von Ressourcen, die mit einer externen Entität geteilt wurden.

Die folgende Liste enthält einige der Dienste, die ressourcenbasierte Richtlinien unterstützen. AWS Eine vollständige Liste der wachsenden Anzahl von AWS Diensten, die das Anhängen von Berechtigungsrichtlinien an Ressourcen statt an Prinzipale unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren und suchen Sie in der Spalte Ressourcenbasiert nach den Diensten, für die Ja angegeben ist.

- Amazon-S3-Buckets – Die Richtlinie ist dem Bucket angefügt, allerdings steuert die Richtlinie sowohl den Zugriff auf den Bucket als auch auf die darin befindlichen Objekte. Weitere Informationen finden Sie unter Access Control im Benutzerhandbuch für Amazon Simple Storage Service. In einigen Fällen kann es sinnvoll sein, Rollen für den kontoübergreifenden Zugriff auf Amazon S3 zu verwenden. Weitere Informationen finden Sie in den Beispiel-Walkthroughs im Benutzerhandbuch für Amazon Simple Storage Service.
- Amazon Simple Notification Service (Amazon SNS)-Themen – Weitere Informationen finden Sie unter Beispiele für die Zugriffskontrolle in Amazon SNS im Entwicklerhandbuch zu Amazon Simple Notification Service.
- Amazon Simple Queue Service (Amazon SQS)-Warteschlangen – Weitere Informationen finden Sie unter Anhang: Sprache der Zugriffsrichtliniensprache im Amazon Simple Queue Service-Entwicklungshandbuch.

Delegieren von AWS Berechtigungen in einer ressourcenbasierten Richtlinie

Wenn eine Ressource den Auftraggeber in Ihrem Konto Berechtigungen gewährt, können Sie diese Berechtigungen dann an bestimmte IAM-Identitäten delegieren. Identitäten sind Benutzer, Benutzergruppen oder Rollen in Ihrem Konto. Sie delegieren Berechtigungen, indem Sie eine

Richtlinie mit der Identität verknüpfen. Sie können maximal die vom ressourcenbesitzenden Konto gewährten Berechtigungen gewähren.

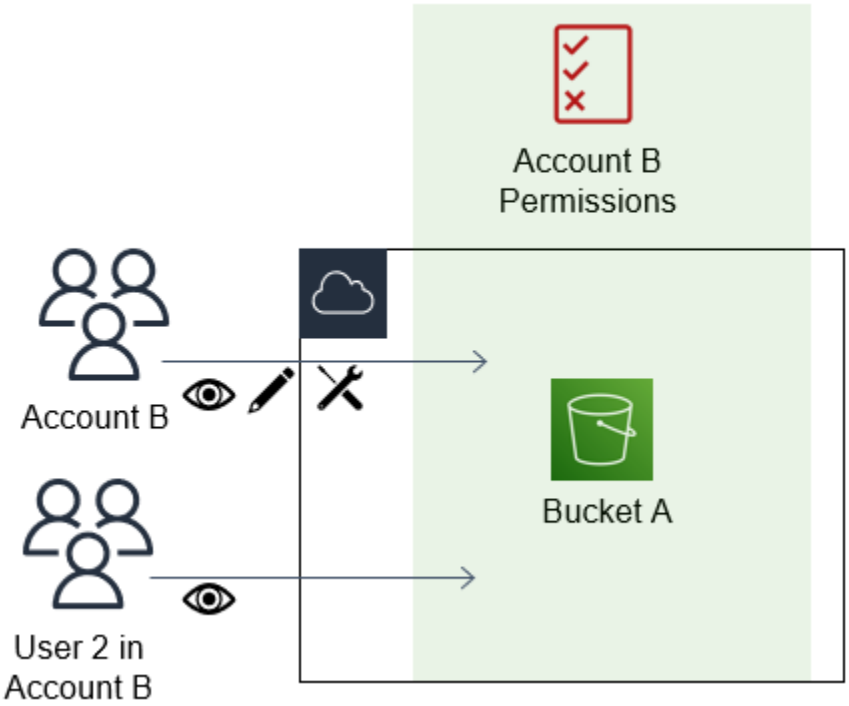
> **Important**
>
> Beim kontoübergreifenden Zugriff benötigt ein Prinzipal `Allow` in der Identitätsrichtlinie und der ressourcenbasierte Richtlinie.

Nehmen Sie an, dass eine ressourcenbasierte Richtlinie allen Auftraggeber in Ihrem Konto vollen administrativen Zugriff auf eine Ressource gewährt. Anschließend können Sie Vollzugriff, Lesezugriff oder beliebigen anderen Teilzugriff auf die Prinzipale in Ihrem Konto delegieren. AWS Wenn die ressourcenbasierte Richtlinie nur Listenberechtigungen gewährt, können Sie alternativ nur den Listenzugriff delegieren. Wenn Sie versuchen, mehr Berechtigungen zu delegieren, als Ihr Konto hat, haben Ihre Auftraggeber immer noch nur Listenzugriff.

Weitere Informationen darüber, wie diese Entscheidungen getroffen werden, finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird.

> **Note**
>
> IAM-Rollen und ressourcenbasierte Richtlinien delegieren den Zugriff auf Konten nur innerhalb einer einzelnen Partition. Beispielsweise können Sie keinen kontenübergreifenden Zugriff zwischen einem Konto in der `aws`-Standardpartition und einem Konto in der `aws-cn`-Partition hinzufügen.

Nehmen Sie beispielsweise an, dass Sie `AccountA` und `AccountB` verwalten. In KontoA haben Sie einen Amazon-S3-Bucket mit dem Namen `BucketA`.



1. Sie weisen `BucketA` eine ressourcenbasierte Richtlinie zu, die allen Prinzipalen in KontoB Vollzugriff auf Objekte in Ihrem Bucket gewährt. Sie können beliebige Objekte in diesem Bucket erstellen, lesen oder löschen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::AccountB:root"},
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
            }
        ]
}
```

KontoA gewährt KontoB durch die Benennung von KontoA als Prinzipal in der ressourcenbasierten Richtlinie Vollzugriff auf BucketA. Dadurch ist KontoB berechtigt, alle Aktionen im Bucket über KontoA auszuführen und der KontoB-Administrator kann seinen Benutzern in KontoB den Zugriff erteilen.

Der Root-Benutzer von KontoB verfügt über alle Berechtigungen, die dem Konto gewährt werden. Daher hat der Root-Benutzer Vollzugriff auf BucketA.

2. In KontoB fügen Sie dem IAM-Benutzer mit dem Namen Benutzer2 eine Richtlinie an. Diese Richtlinie erlaubt dem Benutzer nur Lesezugriff auf die Objekte in BucketA. Das bedeutet, dass Benutzer2 die Objekte anzeigen, aber nicht erstellen, bearbeiten oder löschen kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

Die maximale Zugriffsebene, die KontoB delegieren kann, ist die Zugriffsebene, die dem Konto gewährt wird. In diesem Fall gewährt die ressourcenbasierte Richtlinie KontoB Vollzugriff, aber Benutzer2 wird nur der schreibgeschützte Zugriff gewährt.

Der KontoB-Administrator gewährt Benutzer1 keinen Zugriff. Standardmäßig haben Benutzer keine Berechtigungen, außer denen, die ausdrücklich gewährt werden, sodass Benutzer1 keinen Zugriff auf BucketA hat.

IAM wertet die Berechtigungen eines Auftraggebers zu dem Zeitpunkt aus, zu dem der Auftraggeber eine Anforderung stellt. Wenn Sie Platzhalter (*) verwenden, um Benutzern vollen Zugriff auf Ihre Ressourcen zu gewähren, können Prinzipale auf alle Ressourcen zugreifen, auf die Ihr Konto Zugriff hat. AWS Dies gilt auch für Ressourcen, die Sie nach der Erstellung der Benutzerrichtlinie hinzufügen oder auf die Sie Zugriff erhalten.

Hätte KontoB im vorhergehenden Beispiel eine Richtlinie an Benutzer2 angefügt, die Vollzugriff auf alle Ressourcen in allen Konten gewährt, hätte Benutzer2 automatisch Zugriff auf alle Ressourcen, auf die KontoB Zugriff hat. Dies schließt den BucketA-Zugriff und den Zugriff auf alle anderen Ressourcen ein, die durch ressourcenbasierte Richtlinien in KontoA gewährt werden.

Weitere Hinweise zur komplexen Verwendung von Rollen, z. B. zum Gewähren von Zugriff auf Anwendungen und Services, finden Sie unter Gängige Szenarien für Rollen: Benutzer, Anwendungen und Services.

> **Important**
>
> Gewähren Sie nur Zugriff auf Entitäten, denen Sie vertrauen, und gewähren Sie nur das erforderliche Mindestmaß an Zugriff. Wenn es sich bei der vertrauenswürdigen Entität um ein anderes AWS Konto handelt, kann jedem IAM-Prinzipal Zugriff auf Ihre Ressource gewährt werden. Das vertrauenswürdige AWS Konto kann den Zugriff nur in dem Umfang delegieren, in dem ihm Zugriff gewährt wurde. Es kann nicht mehr Zugriff delegieren, als dem Konto selbst gewährt wurde.

Weitere Informationen über Berechtigungen, Richtlinien und die Sprache der Berechtigungsrichtlinie, mit der Sie die Richtlinien erstellen, finden Sie unter Zugriffsmanagement für AWS Ressourcen.

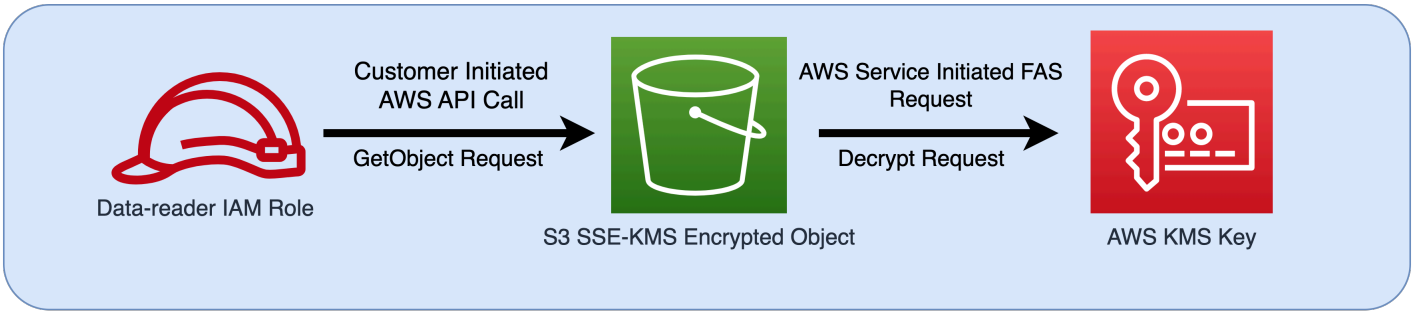
Forward Access Sessions (FAS)

Forward Access Sessions (FAS) ist eine IAM-Technologie, die von AWS Diensten verwendet wird, um Ihre Identität, Berechtigungen und Sitzungsattribute weiterzugeben, wenn ein AWS Service in Ihrem Namen eine Anfrage stellt. FAS verwendet die Berechtigungen der Identität, die einen AWS Dienst aufruft, in Kombination mit der Identität eines AWS Dienstes, um Anfragen an nachgelagerte Dienste zu stellen. FAS-Anfragen werden erst im Namen eines IAM-Prinzipals an AWS Dienste gestellt, nachdem ein Dienst eine Anfrage erhalten hat, für deren Abschluss Interaktionen mit anderen AWS Diensten oder Ressourcen erforderlich sind. Wenn eine FAS-Anfrage gestellt wird:

- Der Service, der die erste Anfrage von einem IAM-Prinzipal empfängt, überprüft die Berechtigungen des IAM-Prinzipals.
- Der Service, der eine nachfolgende FAS-Anfrage empfängt, überprüft auch die Berechtigungen desselben IAM-Prinzipals.

FAS wird beispielsweise von Amazon S3 verwendet, um Aufrufe zur Entschlüsselung eines Objekts AWS Key Management Service zu tätigen, wenn SSE-KMS zur Verschlüsselung verwendet wurde. Beim Herunterladen eines SSE-KMS-verschlüsselten Objekts ruft eine Rolle namens data-reader das Objekt gegen Amazon S3 GetObject auf und ruft es nicht direkt auf. AWS KMS Nach Erhalt der GetObject Anfrage und Autorisierung des Datenlesegeräts stellt Amazon S3 dann eine FAS-Anfrage

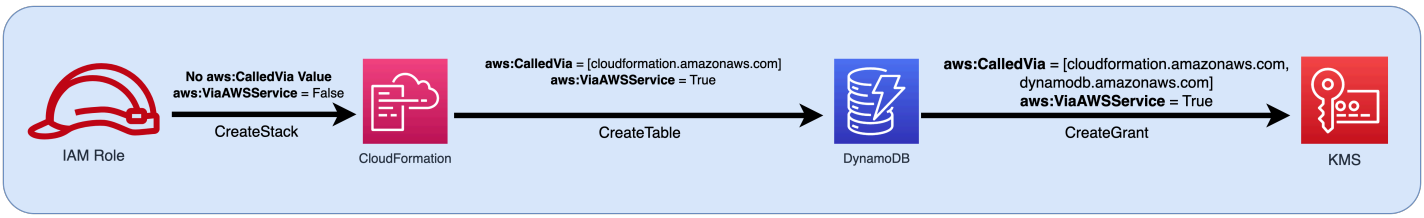
an AWS KMS , um das Amazon S3 S3-Objekt zu entschlüsseln. Wenn KMS die FAS-Anfrage erhält, überprüft KMS die Berechtigungen der Rolle und autorisiert die Entschlüsselungsanforderung nur, wenn „data-reader“ über die richtigen Berechtigungen für den KMS-Schlüssel verfügt. Die Anfragen an Amazon S3 und AWS KMS werden mit den Berechtigungen der Rolle autorisiert und sind nur erfolgreich, wenn der Datenleser über Berechtigungen sowohl für das Amazon S3 S3-Objekt als auch für den AWS KMS-Schlüssel verfügt.



> **Note**
>
> Zusätzliche FAS-Anfragen können von Services gestellt werden, die eine FAS-Anfrage erhalten haben. In solchen Fällen muss der anfordernde Prinzipal über Berechtigungen für alle von FAS aufgerufenen Services verfügen.

FAS-Anfragen und IAM-Richtlinienbedingungen

Wenn FAS-Anfragen gestellt werden, werden aws: CalledVia-, aws: CalledVia Zuerst- und aws: CalledVia Zuletzt-Bedingungsschlüssel mit dem Service-Prinzipal des Service gefüllt, der den FAS-Aufruf initiiert hat. Der Wert des AWS: via AWSService-Bedingungsschlüssels wird immer dann auf `true` gesetzt, wenn eine FAS-Anfrage gestellt wird. In der folgenden Abbildung sind für die CloudFormation direkte Anfrage keine `aws:CalledVia` oder keine `aws:ViaAWSService` Bedingungsschlüssel gesetzt. Wenn CloudFormation DynamoDB im Namen der Rolle nachgelagerte FAS-Anfragen stellen, werden die Werte für diese Bedingungsschlüssel aufgefüllt.



Damit eine FAS-Anfrage gestellt werden kann, die andernfalls durch eine Deny-Richtlinienanweisung mit einem Bedingungsschlüssel, der Quell-IP-Adressen oder Quell-VPCs testet, verweigert werden würde, müssen Sie Bedingungsschlüssel verwenden, um in Ihrer Deny-Richtlinie eine Ausnahme für FAS-Anfragen bereitzustellen. Dies kann für alle FAS-Anfragen mithilfe des `aws:ViaAWSService`-Bedingungsschlüssels erfolgen. Damit nur bestimmte AWS Dienste FAS-Anfragen stellen können, verwenden Sie. `aws:CalledVia`

> **Important**
>
> Wenn eine FAS-Anfrage gestellt wird, nachdem eine erste Anfrage über einen VPC-Endpunkt gestellt wurde, werden die Bedingungsschlüsselwerte für `als: SourceVpce`, `aws: SourceVpc` und `aws: VpcSource Ip` aus der ersten Anfrage nicht in FAS-Anfragen verwendet. Wenn Sie Richtlinien mit `aws:VPCSourceIP` oder `aws:SourceVPCE` schreiben, um bedingten Zugriff zu gewähren, müssen Sie auch `aws:ViaAWSService` oder `aws:CalledVia` verwenden, um FAS-Anfragen zuzulassen. Wenn eine FAS-Anfrage gestellt wird, nachdem eine erste Anfrage bei einem Endpunkt eines öffentlichen AWS Dienstes eingegangen ist, werden nachfolgende FAS-Anfragen mit demselben `aws:SourceIP` Bedingungsschlüsselwert gestellt.

Beispiel: Amazon-S3-Zugriff von einer VPC oder mit FAS zulassen

Im folgenden Beispiel für eine IAM-Richtlinie sind Amazon S3 GetObject - und Athena-Anfragen nur zulässig, wenn sie von VPC-Endpunkten stammen, die an *`example_vpc`* angehängt sind, oder wenn es sich bei der Anfrage um eine FAS-Anfrage von Athena handelt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
```

```
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
            "example_vpc"
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Weitere Beispiele für die Verwendung von Bedingungsschlüsseln, um den FAS-Zugriff zu ermöglichen, finden Sie im Repository Data Perimeter Policy Examples.

Beispiele für identitätsbasierte Richtlinien in IAM

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 IAM-Prinzipal (Benutzer oder Rolle) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden in Form AWS von JSON-Dokumenten gespeichert, die an eine IAM-Identität (Benutzer, Benutzergruppe oder Rolle) angehängt sind. Zu identitätsbasierten Richtlinien gehören verwaltete AWS -Richtlinien, kundenverwaltete Richtlinien und eingebundene Richtlinien. Informationen zum Erstellen einer IAM-Richtlinie mithilfe dieser Beispiel-JSON-Richtliniendokumente finden Sie unter the section called “Erstellen von Richtlinien mit dem JSON-Editor”.

Standardmäßig werden alle Anforderungen verweigert. Daher müssen Sie Zugriffsberechtigungen für alle Services, Aktionen und Ressourcen, auf die die Identität zugreifen soll, erteilen. Wenn Sie auch

den Zugriff zur Durchführung der angegebenen Aktionen in der IAM-Konsole gewähren möchten, müssen Sie zusätzliche Berechtigungen erteilen.

Nachstehende Richtlinienbibliothek hilft Ihnen bei der Definition der Berechtigungen für Ihre IAM-Identitäten. Nachdem Sie die benötigte Richtlinie gefunden haben, klicken Sie auf View this policy (Richtlinie anzeigen), um den JSON-Code der Richtlinie anzuzeigen. Sie können das JSON-Richtliniendokument als Vorlage für Ihre eigenen Richtlinien verwenden.

> **Note**
>
> Wenn Sie eine Richtlinie zur Aufnahme in dieses Referenzhandbuch vorschlagen möchten, verwenden Sie die Schaltfläche Feedback unten auf der Seite.

Beispielrichtlinien: AWS

- Gewährt den Zugriff innerhalb eines bestimmten Datumsbereichs. (Richtlinie anzeigen.)
- Ermöglicht das Aktivieren und Deaktivieren von AWS Regionen. (Richtlinie anzeigen.)
- Ermöglicht MFA-authentifizierten Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten. (Richtlinie anzeigen.)
- Gewährt spezifischen Zugriff bei der Verwendung von MFA in einem bestimmten Datumsbereich. (Richtlinie anzeigen.)
- Ermöglicht Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten. (Richtlinie anzeigen.)
- Ermöglicht Benutzern, ihr eigenes MFA-Gerät auf der Seite Sicherheitsanmeldedaten zu verwalten. (Richtlinie anzeigen.)
- Ermöglicht Benutzern, ihr eigenes Passwort auf der Seite mit den Sicherheitsanmeldedaten zu verwalten. (Richtlinie anzeigen.)
- Ermöglicht Benutzern, ihr eigenes Passwort, ihre eigenen Zugangsschlüssel und öffentlichen SSH-Schlüssel auf der Seite mit den Sicherheitsanmeldedaten zu verwalten. (Richtlinie anzeigen.)
- Verweigert den Zugriff auf AWS basierend auf der angeforderten Region. (Richtlinie anzeigen.)
- Verweigert den Zugriff auf AWS basierend auf der Quell-IP-Adresse. (Richtlinie anzeigen.)

Beispiel für eine Richtlinie: AWS Data Exchange

- Verweigern Sie den Zugriff auf Amazon-S3-Ressourcen außerhalb Ihres Kontos, mit Ausnahme von AWS Data Exchange. (Richtlinie anzeigen.)

Beispielrichtlinien: AWS Data Pipeline

- Verweigert den Zugriff auf nicht vom Benutzer erstellte Pipelines (Richtlinie anzeigen).

Beispielrichtlinien: Amazon DynamoDB

- Ermöglicht den Zugriff auf eine bestimmte Amazon DynamoDB-Tabelle (Richtlinie anzeigen.)
- Ermöglicht den Zugriff auf bestimmte Amazon DynamoDB-Attribute (Richtlinie anzeigen).
- Ermöglicht den zeilenweisen Zugriff auf Amazon DynamoDB, basierend auf einer Amazon Cognito ID-ID (Richtlinie anzeigen.)

Beispiele für Richtlinien: Amazon EC2

- Ermöglicht das anfügen oder Entfernen von Amazon EBS-Volumes an Amazon EC2-Instances auf der Grundlage von Tags (Diese Richtlinie anzeigen.)
- Ermöglicht es, Amazon EC2-Instances in einem bestimmten Subnetz programmgesteuert und in der Konsole zu starten (Richtlinie anzeigen)
- Ermöglicht es, Amazon EC2-Sicherheitsgruppen, die mit einem bestimmten VPC verknüpft sind, programmgesteuert und in der Konsole zu verwalten (Richtlinie anzeigen).
- Ermöglicht es, Amazon EC2-Instances, die ein Benutzer markiert hat, programmgesteuert und in der Konsole zu starten oder zu stoppen (Richtlinie anzeigen).
- Ermöglicht das Starten oder Anhalten von Amazon EC2-Instances, basierend auf Ressourcen- und Auftraggeber-Tags, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht das Starten oder Anhalten von Amazon EC2-Instances, wenn die Ressourcen- und Auftraggeber-Tags übereinstimmen (Richtlinie anzeigen).
- Ermöglicht vollen Amazon EC2-Zugriff innerhalb einer bestimmten Region, programmatisch und in der Konsole. (Richtlinie anzeigen.)
- Ermöglicht es, programmgesteuert und in der Konsole eine bestimmte Amazon EC2-Instance zu starten und zu stoppen und eine spezifischen Sicherheitsgruppe zu ändern (Richtlinie anzeigen)

- Verweigert den Zugriff auf bestimmte Amazon EC2-Operationen ohne MFA (Richtlinie anzeigen).
- Schränkt das Beenden einer Amazon EC2-Instance auf einen bestimmten IP-Adressbereich ein (Richtlinie anzeigen)

Beispielrichtlinien: AWS Identity and Access Management (IAM)

- Ermöglicht den Zugriff auf die Richtliniensimulator-API (Richtlinie anzeigen).
- Ermöglicht den Zugriff auf die Richtliniensimulator-Konsole (Richtlinie anzeigen).
- Ermöglicht die Annahme aller Rollen mit einem bestimmten Tag, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht und verweigert den Zugriff auf mehrere Services, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht das Hinzufügen eines spezifischen Tags an einen IAM-Benutzer mit einem anderen spezifischen Tag, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht das Hinzufügen eines spezifischen Tags zu einem bzw. einer beliebigen IAM-Benutzer oder -Rolle, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht das Erstellen eines neuen Benutzers nur mit bestimmten Tags (Richtlinie anzeigen)
- Ermöglicht das Generieren und Abrufen von IAM-Berichten zu den Anmeldeinformationen (Richtlinie anzeigen).
- Ermöglicht die Verwaltung einer Gruppenmitgliedschaft, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht das Verwalten eines bestimmten Tags (Richtlinie anzeigen.)
- Ermöglicht die Übergabe einer IAM-Rolle an einen bestimmten Service (Richtlinie anzeigen).
- Ermöglicht schreibgeschützten Zugriff auf die IAM-Konsole ohne Berichterstattung (Richtlinie anzeigen).
- Ermöglicht schreibgeschützten Zugriff auf die IAM-Konsole (Richtlinie anzeigen).
- Ermöglicht spezifischen Benutzern die Verwaltung einer Gruppe, programmgesteuert und in der Konsole (Richtlinie anzeigen).
- Ermöglicht das Festlegen der Kontopasswortanforderungen, programmgesteuert und in der Konsole (Richtlinie anzeigen.)
- Ermöglicht den Benutzern mit einem bestimmten Pfad die Verwendung der Richtliniensimulator-API (Richtlinie anzeigen).

- Ermöglicht den Benutzern mit einem bestimmten Pfad die Verwendung der Richtliniensimulator-Konsole (Richtlinie anzeigen).
- IAM: Ermöglicht es IAM-Benutzern, MFA-Geräte selbst zu verwalten (Richtlinie anzeigen.)
- Ermöglicht IAM-Benutzern, ihre eigenen Anmeldeinformationen programmgesteuert und in der Konsole festzulegen. (Richtlinie anzeigen.)
- Ermöglicht das Anzeigen von Informationen über den Dienst, auf den zuletzt zugegriffen wurde, für eine AWS Organizations Richtlinie in der IAM-Konsole. (Richtlinie anzeigen.)
- Beschränkt die verwalteten Richtlinien, die IAM-Benutzern, -Gruppen oder -Rollen zugeordnet werden können (Richtlinie anzeigen).
- Erlaubt den Zugriff auf IAM-Richtlinien nur in Ihrem Konto (Diese Richtlinie anzeigen.)

Beispielrichtlinien: AWS Lambda

- Ermöglicht einer AWS Lambda Funktion den Zugriff auf eine Amazon DynamoDB-Tabelle (Diese Richtlinie anzeigen.)

Beispielrichtlinien: Amazon RDS

- Ermöglicht den vollständigen Amazon RDS-Datenbankzugriff innerhalb einer bestimmten Region. (Richtlinie anzeigen.)
- Ermöglicht es, Amazon RDS-Datenbanken programmgesteuert und in der Konsole wiederherzustellen (Richtlinie anzeigen)
- Gewährt Tag-Eigentümern Vollzugriff auf die von ihnen markierten Amazon RDS-Ressourcen (Richtlinie anzeigen)

Beispiele für Richtlinien: Amazon S3

- Ermöglicht es einem Amazon Cognito-Benutzer, auf Objekte in seinem eigenen Amazon S3-Bucket zuzugreifen (Richtlinie anzeigen)
- Ermöglicht Verbundbenutzern den Zugriff auf ihr eigenes Home-Verzeichnis in Amazon S3, programmatisch und in der Konsole (Diese Richtlinie anzeigen.)
- Ermöglicht vollständigen S3-Zugriff. Verweigert jedoch explizit den Zugriff auf den Produktions-Bucket, wenn der Administrator sich nicht innerhalb der letzten dreißig Minuten mit MFA angemeldet hat, (Richtlinie anzeigen).

- Ermöglicht es IAM-Benutzern, programmgesteuert und in der Konsole auf ihr eigenes Stammverzeichnis in Amazon S3 zuzugreifen (Richtlinie anzeigen)
- Ermöglicht einem Benutzer die Verwaltung eines einzelnen Amazon S3 S3-Buckets und verweigert alle anderen AWS Aktionen und Ressourcen (Diese Richtlinie anzeigen.)
- Ermöglicht den `Read`- und `Write`-Zugriff auf einen bestimmten Amazon S3-Bucket (Richtlinie anzeigen)
- Ermöglicht den `Read`- und `Write`-Zugriff programmgesteuert und in der Konsole auf einen bestimmten Amazon S3-Bucket (Richtlinie anzeigen)

AWS: Ermöglicht Zugriff basierend auf Datum und Uhrzeit

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den Zugriff auf Aktionen basierend auf Datum und Uhrzeit erlaubt. Diese Richtlinie beschränkt den Zugriff auf Aktionen, die zwischen dem 1. April 2020 und dem 30. Juni 2020 (UTC) stattfinden. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Weitere Informationen zum Verwenden mehrerer Bedingungen innerhalb eines `Condition`-Blocks einer IAM-Richtlinie finden Sie unter Mehrere Werte in einer Bedingung.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> Sie können eine Richtlinienvariable nicht mit dem Datum-Bedingungsoperator verwenden. Weitere Informationen finden Sie unter Condition-Element

AWS: Ermöglicht das Aktivieren und Deaktivieren von Regionen AWS

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es einem Administrator erlaubt, die Region Asien-Pazifik (Hongkong) (ap-east-1) zu aktivieren und zu deaktivieren. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Diese Einstellung wird auf der Seite Account Settings (Kontoeinstellungen) in der AWS Management Console angezeigt. Diese Seite enthält sensible Informationen auf Kontoebene, die nur von Kontoadministratoren angezeigt und verwaltet werden sollten. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

> **Important**
>
> Standardmäßig aktivierte Regionen können weder aktiviert noch deaktiviert werden. Sie können nur Regionen einschließen, die standardmäßig deaktiviert sind. Weitere Informationen finden Sie unter Verwalten von AWS -Regionen im Allgemeine AWS-Referenz.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
```

```
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es IAM-Benutzern, die mithilfe der Multi-Faktor-Authentifizierung (MFA) authentifiziert wurden, ermöglicht, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten. Diese Seite der AWS Management Console zeigt Kontoinformationen wie z. B. die Konto-ID und die kanonische Benutzer-ID an. Benutzer können auch ihre eigenen Passwörter, Zugriffsschlüssel, MFA-Geräte, X.509-Zertifikate und SSH-Schlüssel und Git-Anmeldeinformationen anzeigen und bearbeiten. Diese Beispielrichtlinie enthält die erforderlichen Berechtigungen zum Anzeigen und Bearbeiten aller Informationen auf der Seite. Außerdem muss der Benutzer MFA einrichten und authentifizieren, bevor er andere Operationen in ausführt. AWS Informationen darüber, wie Benutzern die Verwaltung ihrer eigenen Anmeldeinformationen ohne MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Informationen darüber, wie Benutzer auf die Seite mit den Sicherheitsanmeldedaten zugreifen können, finden Sie unter. Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole)

> **Note**
>
> - Diese Beispielrichtlinie erlaubt es Benutzern nicht, ein Passwort zurückzusetzen, wenn sie sich AWS Management Console zum ersten Mal anmelden. Wir empfehlen, dass Sie neuen Benutzern keine Berechtigungen erteilen, bis sie sich angemeldet haben. Weitere Informationen finden Sie unter Wie erstelle ich sicher IAM-Benutzer?. Dies verhindert auch, dass Benutzer mit einem abgelaufenen Passwort ihr Passwort während der Anmeldung zurücksetzen können. Sie können dies erlauben, indem Sie `iam:ChangePassword` und `iam:GetAccountPasswordPolicy` der Anweisung `DenyAllExceptListedIfNoMFA`

hinzufügen. Wir empfehlen dies jedoch nicht, da es ein Sicherheitsrisiko darstellen kann, Benutzern eine Änderung ihres Passworts ohne MFA zu erlauben.

- Wenn Sie diese Richtlinie für den programmatischen Zugriff verwenden möchten, müssen Sie `GetSessionToken` anrufen, um sich bei MFA zu authentifizieren. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs.

Was macht diese Richtlinie?

- Die `AllowViewAccountInfo`-Anweisung ermöglicht es dem Benutzer, Informationen auf Kontoebene anzuzeigen. Diese Berechtigungen müssen in ihrer eigenen Anweisung vorliegen, da sie keine Ressourcen-ARN unterstützen oder keine angeben müssen. Geben Sie anstelle von Berechtigungen `"Resource" : "*"` ein. Diese Anweisung enthält die folgenden Aktionen, mit denen der Benutzer spezifische Informationen anzeigen kann:
  - `GetAccountPasswordPolicy` – Zeigt die Passwortanforderungen des Kontos beim Ändern seines eigenen IAM-Benutzer-Passworts an.
  - `ListVirtualMFADevices` – Zeigt Details über ein virtuelles MFA-Gerät an, das für den Benutzer aktiviert ist.
- Mit der `AllowManageOwnPasswords`-Anweisung kann der Benutzer sein eigenes Passwort ändern. Diese Anweisung enthält auch die `GetUser`-Aktion, die erforderlich ist, um die meisten Informationen auf der Seite My Security Credentials (Meine Sicherheitsanmeldeinformationen) anzuzeigen.
- Die `AllowManageOwnAccessKeys`-Anweisung ermöglicht dem Benutzer das Erstellen, Aktualisieren und Löschen seiner eigenen Zugriffsschlüssel. Der Benutzer kann auch Informationen darüber abrufen, wann der angegebene Zugriffsschlüssel zuletzt verwendet wurde.
- Die `AllowManageOwnSigningCertificates`-Anweisung ermöglicht dem Benutzer das Hochladen, Aktualisieren und Löschen seiner eigenen Signaturzertifikate.
- Die `AllowManageOwnSSHPublicKeys` Anweisung ermöglicht es dem Benutzer, seine eigenen öffentlichen SSH-Schlüssel für CodeCommit hochzuladen, zu aktualisieren und zu löschen.
- Die `AllowManageOwnGitCredentials` Anweisung ermöglicht es dem Benutzer, seine eigenen Git-Anmeldeinformationen für zu erstellen, zu aktualisieren und zu löschen CodeCommit.
- Die `AllowManageOwnVirtualMFADevice`-Anweisung ermöglicht dem Benutzer das Erstellen seines eigenen virtuellen MFA-Geräts. Der Ressourcen-ARN in dieser Anweisung erlaubt es dem Benutzer, ein MFA-Gerät mit einem beliebigen Namen zu erstellen, aber die anderen Anweisungen

in der Richtlinie erlauben es dem Benutzer nur, das Gerät mit dem aktuell angemeldeten Benutzer zu verknüpfen.

- Die `AllowManageOwnUserMFA`-Anweisung ermöglicht es dem Benutzer, das virtuelle, U2F- oder physische MFA-Gerät für seinen eigenen Benutzer anzuzeigen oder zu verwalten. Der Ressourcen-ARN in dieser Anweisung gewährt nur Zugriff auf den eigenen IAM-Benutzer des Benutzers. Benutzer können das MFA-Gerät für andere Benutzer nicht anzeigen oder verwalten.
- Die `DenyAllExceptListedIfNoMFA` Anweisung verweigert den Zugriff auf alle Aktionen in allen AWS Diensten, mit Ausnahme einiger aufgelisteter Aktionen, jedoch nur, wenn der Benutzer nicht mit MFA angemeldet ist. Die Anweisung verwendet eine Kombination aus `"Deny"` und `"NotAction"`, um den Zugriff auf alle Aktionen, die nicht aufgeführt sind, explizit zu verwehren. Die aufgeführten Elemente werden durch diese Anweisung nicht verweigert oder zugelassen. Die Aktionen werden aber durch andere Anweisungen in der Richtlinie zugelassen. Weitere Informationen zur Logik dieser Anweisung finden Sie unter NotAction with Deny. Hat sich der Benutzer mit MFA angemeldet, schlägt der `Condition`-Test fehl und diese Anweisung verwehrt keine Aktionen. In diesem Fall bestimmen andere Richtlinien oder Anweisungen für den Benutzer die Berechtigungen des Benutzers.

  Durch diese Anweisung wird sichergestellt, dass der Benutzer, wenn er nicht mit MFA angemeldet ist, nur die aufgeführten Aktionen ausführen kann. Außerdem kann er die aufgelisteten Aktionen nur ausführen, wenn eine andere Anweisung oder Richtlinie den Zugriff auf diese Aktionen gewährt. Dies ermöglicht es einem Benutzer nicht, beim Anmelden ein Passwort zu erstellen, da die `iam:ChangePassword`-Aktion nicht ohne MFA-Autorisierung erlaubt sein sollte.

  Die `...IfExists`-Version des `Bool` -Operators stellt sicher, dass bei fehlendem `als: MultiFactor AuthPresent`-Schlüssel die Bedingung den Wert "True" zurückgibt. Dies bedeutet, dass einem Benutzer, der mit langfristigen Anmeldeinformationen auf eine API zugreift, wie z. B. einem Zugriffsschlüssel, der Zugriff auf die Nicht-IAM-API-Operationen verweigert wird.

Diese Richtlinie ermöglicht es Benutzern nicht, die Seite Users (Benutzer) in der IAM-Konsole anzuzeigen oder diese Seite für den Zugriff auf ihre eigenen Benutzerinformationen zu verwenden. Um dies zuzulassen, fügen Sie die Aktion `iam:ListUsers` der Anweisung `AllowViewAccountInfo` und der Anweisung `DenyAllExceptListedIfNoMFA` hinzu. Darüber hinaus ermöglicht sie Benutzern nicht, ihr Passwort auf ihrer eigenen Benutzerseite anzupassen. Um dies zu ermöglichen, fügen Sie die Aktionen `iam:GetLoginProfile` und `iam:UpdateLoginProfile` zur Anweisung `AllowManageOwnPasswords` hinzu. Um es einem Benutzer auch zu ermöglichen, sein Passwort über seine eigene Benutzerseite zu ändern, ohne

sich mit MFA anzumelden, fügen Sie die Aktion `iam:UpdateLoginProfile` zur Anweisung `DenyAllExceptListedIfNoMFA` hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
```

```
            "iam:UploadSigningCertificate"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSSHPublicKeys",
        "Effect": "Allow",
        "Action": [
            "iam:DeleteSSHPublicKey",
            "iam:GetSSHPublicKey",
            "iam:ListSSHPublicKeys",
            "iam:UpdateSSHPublicKey",
            "iam:UploadSSHPublicKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnGitCredentials",
        "Effect": "Allow",
        "Action": [
            "iam:CreateServiceSpecificCredential",
            "iam:DeleteServiceSpecificCredential",
            "iam:ListServiceSpecificCredentials",
            "iam:ResetServiceSpecificCredential",
            "iam:UpdateServiceSpecificCredential"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnVirtualMFADevice",
        "Effect": "Allow",
        "Action": [
            "iam:CreateVirtualMFADevice"
        ],
        "Resource": "arn:aws:iam::*:mfa/*"
    },
    {
        "Sid": "AllowManageOwnUserMFA",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice",
            "iam:EnableMFADevice",
            "iam:ListMFADevices",
            "iam:ResyncMFADevice"
```

```
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

AWS: Ermöglicht spezifischen Zugriff mit MFA innerhalb bestimmter Daten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen, die mehrere Bedingungen verwendet, die mithilfe von einem logischen AND bewertet werden. Es ermöglicht vollständigen Zugriff auf den Service mit dem Namen `SERVICE-NAME-1`und den Zugriff auf die Aktionen `ACTION-NAME-A` und `ACTION-NAME-B` im Service mit dem Namen `SERVICE-NAME-2`. Diese Aktionen sind nur zulässig, wenn der Benutzer mit der Multi-Factor Authentication (MFA) authentifiziert wird. Der Zugriff ist auf Aktionen vom 1. Juli 2017 bis einschließlich 31. Dezember 2017 (UTC) beschränkt. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Weitere Informationen zum Verwenden mehrerer Bedingungen innerhalb eines `Condition`-Blocks einer IAM-Richtlinie finden Sie unter Mehrere Werte in einer Bedingung.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS: Ermöglicht IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es IAM-Benutzern ermöglicht, all ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten. AWS Management Console Auf dieser Seite werden Kontoinformationen wie die Konto-ID und die kanonische Benutzer-ID angezeigt. Benutzer können auch ihre eigenen Passwörter, Zugriffsschlüssel, X.509-Zertifikate, SSH-Schlüssel und Git-Anmeldeinformationen anzeigen und bearbeiten. Diese Beispielrichtlinie enthält die erforderlichen Berechtigungen zum Anzeigen und Bearbeiten aller Informationen auf der Seite mit Ausnahme des MFA-Geräts des Benutzers. Informationen darüber, wie Benutzern die Verwaltung ihrer eigenen Anmeldeinformationen mit MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Informationen darüber, wie Benutzer auf die Seite mit den Sicherheitsanmeldedaten zugreifen können, finden Sie unter. Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole)

Was macht diese Richtlinie?

- Die `AllowViewAccountInfo`-Anweisung ermöglicht es dem Benutzer, Informationen auf Kontoebene anzuzeigen. Diese Berechtigungen müssen in ihrer eigenen Anweisung vorliegen, da sie keine Ressourcen-ARN unterstützen oder keine angeben müssen. Geben Sie anstelle von Berechtigungen `"Resource" : "*"` ein. Diese Anweisung enthält die folgenden Aktionen, mit denen der Benutzer spezifische Informationen anzeigen kann:
  - `GetAccountPasswordPolicy` – Zeigt die Passwortanforderungen des Kontos beim Ändern seines eigenen IAM-Benutzer-Passworts an.
  - `GetAccountSummary` – Zeigt die Konto-ID und die kanonische Benutzer-ID des Kontos an.
- Mit der `AllowManageOwnPasswords`-Anweisung kann der Benutzer sein eigenes Passwort ändern. Diese Anweisung enthält auch die `GetUser`-Aktion, die erforderlich ist, um die meisten Informationen auf der Seite My Security Credentials (Meine Sicherheitsanmeldeinformationen) anzuzeigen.
- Die `AllowManageOwnAccessKeys`-Anweisung ermöglicht dem Benutzer das Erstellen, Aktualisieren und Löschen seiner eigenen Zugriffsschlüssel. Der Benutzer kann auch Informationen darüber abrufen, wann der angegebene Zugriffsschlüssel zuletzt verwendet wurde.
- Die `AllowManageOwnSigningCertificates`-Anweisung ermöglicht dem Benutzer das Hochladen, Aktualisieren und Löschen seiner eigenen Signaturzertifikate.
- Die `AllowManageOwnSSHPublicKeys` Anweisung ermöglicht es dem Benutzer, seine eigenen öffentlichen SSH-Schlüssel für CodeCommit hochzuladen, zu aktualisieren und zu löschen.
- Die `AllowManageOwnGitCredentials` Anweisung ermöglicht es dem Benutzer, seine eigenen Git-Anmeldeinformationen für zu erstellen, zu aktualisieren und zu löschen CodeCommit.

Diese Richtlinie erlaubt es Benutzern nicht, ihre eigenen MFA-Geräte anzuzeigen oder zu verwalten. Außerdem können sie die Seite Users (Benutzer) in der IAM-Konsole nicht anzuzeigen oder für den Zugriff auf ihre eigenen Benutzerinformationen verwenden. Um dies zu ermöglichen, fügen Sie die Aktion `iam:ListUsers` zur Anweisung `AllowViewAccountInfo` hinzu. Darüber hinaus ermöglicht sie Benutzern nicht, ihr Passwort auf ihrer eigenen Benutzerseite anzupassen. Um dies zu ermöglichen, fügen Sie die Aktionen `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile` und `iam:UpdateLoginProfile` zur Anweisung `AllowManageOwnPasswords` hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
```

```
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihr eigenes MFA-Gerät auf der Seite mit den Sicherheitsanmeldedaten zu verwalten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es IAM-Benutzern, die über Multi-Factor Authentication (MFA) authentifiziert wurden, ermöglicht, ihr eigenes MFA-Gerät auf der Seite Sicherheitsanmeldedaten zu verwalten. AWS Management Console Auf dieser Seite werden Konto- und Benutzerinformationen angezeigt, aber der Benutzer kann nur sein eigenes MFA-Gerät anzeigen und bearbeiten. Informationen darüber, wie Benutzern die Verwaltung aller ihrer eigenen Anmeldeinformationen mit MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

> **Note**
>
> Wenn ein IAM-Benutzer mit dieser Richtlinie nicht MFA-authentifiziert ist, verweigert diese Richtlinie den Zugriff auf alle AWS Aktionen außer denen, die für die Authentifizierung mit MFA erforderlich sind. Um die AWS CLI AWS AND-API verwenden zu können, müssen IAM-Benutzer zuerst ihr MFA-Token mithilfe des AWS STS GetSessionTokenVorgangs abrufen und dann dieses Token verwenden, um den gewünschten Vorgang zu authentifizieren.

Andere Richtlinien, wie zum Beispiel ressourcenbasierte Richtlinien oder andere identitätsbasierte Richtlinien, können Aktionen in anderen Services erlauben. Diese Richtlinie verweigert den Zugriff, wenn der IAM-Benutzer nicht per MFA authentifiziert wurde.

Informationen darüber, wie Benutzer auf die Seite mit den Sicherheitsanmeldedaten zugreifen können, finden Sie unter. Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole)

Was macht diese Richtlinie?

- Die Anweisung `AllowViewAccountInfo` erlaubt dem Benutzer das Anzeigen von Details zu einem virtuellen MFA-Gerät, das für den Benutzer aktiviert ist. Diese Berechtigung muss als eigene Anweisung vorliegen, da sie nicht die Angabe eines Ressourcen-ARN unterstützt. Stattdessen müssen Sie `"Resource" : "*"` angeben.
- Die `AllowManageOwnVirtualMFADevice`-Anweisung ermöglicht dem Benutzer das Erstellen seines eigenen virtuellen MFA-Geräts. Der Ressourcen-ARN in dieser Anweisung erlaubt es dem Benutzer, ein MFA-Gerät mit einem beliebigen Namen zu erstellen, aber die anderen Anweisungen in der Richtlinie erlauben es dem Benutzer nur, das Gerät mit dem aktuell angemeldeten Benutzer zu verknüpfen.
- Die `AllowManageOwnUserMFA`-Anweisung ermöglicht es dem Benutzer, sein eigenes virtuelles, U2F- oder physisches MFA-Gerät anzuzeigen oder zu verwalten. Der Ressourcen-ARN in dieser Anweisung gewährt nur Zugriff auf den eigenen IAM-Benutzer des Benutzers. Benutzer können das MFA-Gerät für andere Benutzer nicht anzeigen oder verwalten.
- Die `DenyAllExceptListedIfNoMFA` Anweisung verweigert den Zugriff auf alle Aktionen in allen AWS Diensten, mit Ausnahme einiger aufgelisteter Aktionen, jedoch nur, wenn der Benutzer nicht mit MFA angemeldet ist. Die Anweisung verwendet eine Kombination aus `"Deny"` und `"NotAction"`, um den Zugriff auf alle Aktionen, die nicht aufgeführt sind, explizit zu verwehren. Die aufgeführten Elemente werden durch diese Anweisung nicht verweigert oder zugelassen. Die Aktionen werden aber durch andere Anweisungen in der Richtlinie zugelassen. Weitere Informationen zur Logik dieser Anweisung finden Sie unter NotAction with Deny. Hat sich der Benutzer mit MFA angemeldet, schlägt der `Condition`-Test fehl und diese Anweisung verwehrt keine Aktionen. In diesem Fall bestimmen andere Richtlinien oder Anweisungen für den Benutzer die Berechtigungen des Benutzers.

  Durch diese Anweisung wird sichergestellt, dass der Benutzer, wenn er nicht mit MFA angemeldet ist, nur die aufgeführten Aktionen ausführen kann. Außerdem kann er die aufgelisteten Aktionen

nur ausführen, wenn eine andere Anweisung oder Richtlinie den Zugriff auf diese Aktionen gewährt.

Die `...IfExists`-Version des `Bool` -Operators stellt sicher, dass bei fehlendem `aws:MultiFactorAuthPresent`-Schlüssel die Bedingung den Wert "True" zurückgibt. Dies bedeutet, dass einem Benutzer, der mit langfristigen Anmeldeinformationen auf eine API-Operation zugreift, wie z. B. einem Zugriffsschlüssel, der Zugriff auf die Nicht-IAM-API-Operationen verweigert wird.

Diese Richtlinie ermöglicht es Benutzern nicht, die Seite Users (Benutzer) in der IAM-Konsole anzuzeigen oder diese Seite für den Zugriff auf ihre eigenen Benutzerinformationen zu verwenden. Um dies zuzulassen, fügen Sie die Aktion `iam:ListUsers` der Anweisung `AllowViewAccountInfo` und der Anweisung `DenyAllExceptListedIfNoMFA` hinzu.

> **Warning**
>
> Erlauben Sie nicht das Löschen eines MFA-Geräts ohne MFA-Authentifizierung. Benutzer mit dieser Richtlinie könnten möglicherweise versuchen, sich selbst ein virtuelles MFA-Gerät zuzuweisen und eine Fehlermeldung erhalten, da sie zur Ausführung von `iam:DeleteVirtualMFADevice` nicht berechtigt sind. Wenn dies der Fall ist, fügen Sie diese Berechtigung nicht zur `DenyAllExceptListedIfNoMFA`-Anweisung hinzu. Benutzer, die nicht über MFA authentifiziert wurden, sollten niemals zum Löschen Ihres MFA-Geräts berechtigt werden. Benutzern könnte diese Fehlermeldung angezeigt werden, wenn sie damit begonnen haben, ein virtuelles MFA-Gerät zu ihrem Benutzer zuzuweisen und den Prozess abgebrochen haben. Um dieses Problem zu beheben, müssen Sie oder ein anderer Administrator das vorhandene virtuelle MFA-Gerät des Benutzers mithilfe der AWS API AWS CLI oder löschen. Weitere Informationen finden Sie unter Ich bin nicht berechtigt, Folgendes auszuführen: iam: MFADevice DeleteVirtual.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
```

```
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS: Ermöglicht IAM-Benutzern, ihr eigenes Konsolenkennwort auf der Seite mit den Sicherheitsanmeldedaten zu ändern

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es IAM-Benutzern ermöglicht, ihr eigenes AWS Management Console Passwort auf der Seite Sicherheitsanmeldedaten zu ändern. AWS Management Console Auf dieser Seite werden Konto- und Benutzerinformationen angezeigt, aber der Benutzer kann nur auf sein eigenes Passwort zugreifen. Informationen darüber, wie Benutzern die Verwaltung aller ihrer eigenen Anmeldeinformationen mit MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten. Informationen darüber, wie Benutzern die Verwaltung ihrer eigenen Anmeldeinformationen ohne MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Informationen darüber, wie Benutzer auf die Seite mit den Sicherheitsanmeldedaten zugreifen können, finden Sie unterWie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole).

Was macht diese Richtlinie?

- Die Anweisung `ViewAccountPasswordRequirements` ermöglicht es dem Benutzer, die Passwortanforderungen des Kontos beim Ändern seines eigenen IAM-Benutzer-Passworts anzuzeigen.
- Mit der `ChangeOwnPassword`-Anweisung kann der Benutzer sein eigenes Passwort ändern. Diese Anweisung enthält auch die `GetUser`-Aktion, die erforderlich ist, um die meisten Informationen auf der Seite My Security Credentials (Meine Sicherheitsanmeldeinformationen) anzuzeigen.

Diese Richtlinie ermöglicht es Benutzern nicht, die Seite Users (Benutzer) in der IAM-Konsole anzuzeigen oder diese Seite für den Zugriff auf ihre eigenen Benutzerinformationen zu verwenden. Um dies zu ermöglichen, fügen Sie die Aktion `iam:ListUsers` zur Anweisung `ViewAccountPasswordRequirements` hinzu. Darüber hinaus ermöglicht sie Benutzern nicht, ihr Passwort auf ihrer eigenen Benutzerseite anzupassen. Um dies zu ermöglichen, fügen Sie die Aktionen `iam:GetLoginProfile` und `iam:UpdateLoginProfile` zur Anweisung `ChangeOwnPasswords` hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Ermöglicht IAM-Benutzern, ihr eigenes Passwort, ihre eigenen Zugangsschlüssel und öffentlichen SSH-Schlüssel auf der Seite Sicherheitsanmeldedaten zu verwalten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es IAM-Benutzern ermöglicht, ihr eigenes Passwort, ihre Zugriffsschlüssel und X.509-Zertifikate auf der Seite Sicherheitsanmeldedaten zu verwalten. Diese Seite der AWS Management Console zeigt Kontoinformationen wie z. B. die Konto-ID und die kanonische Benutzer-ID an. Benutzer können auch ihre eigenen Passwörter, Zugriffsschlüssel, MFA-Geräte, X.509-Zertifikate, SSH-Schlüssel und Git-Anmeldeinformationen anzeigen und bearbeiten. Diese Beispielrichtlinie enthält die Berechtigungen, die erforderlich sind, um nur ihr Passwort, ihre Zugriffsschlüssel und ihr X.509-Zertifikat anzuzeigen. Informationen darüber, wie Benutzern die Verwaltung aller ihrer eigenen Anmeldeinformationen mit MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten. Informationen darüber, wie Benutzern die Verwaltung ihrer eigenen Anmeldeinformationen ohne MFA ermöglicht wird, finden Sie unter AWS: Ermöglicht IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Informationen darüber, wie Benutzer auf die Seite mit den Sicherheitsanmeldedaten zugreifen können, finden Sie unter. Wie IAM-Benutzer ihr eigenes Passwort ändern können (Konsole)

Was macht diese Richtlinie?

- Die `AllowViewAccountInfo`-Anweisung ermöglicht es dem Benutzer, Informationen auf Kontoebene anzuzeigen. Diese Berechtigungen müssen in ihrer eigenen Anweisung vorliegen, da sie keine Ressourcen-ARN unterstützen oder keine angeben müssen. Geben Sie anstelle von Berechtigungen `"Resource" : "*"` ein. Diese Anweisung enthält die folgenden Aktionen, mit denen der Benutzer spezifische Informationen anzeigen kann:
  - `GetAccountPasswordPolicy` – Zeigt die Passwortanforderungen des Kontos beim Ändern seines eigenen IAM-Benutzer-Passworts an.
  - `GetAccountSummary` – Zeigt die Konto-ID und die kanonische Benutzer-ID des Kontos an.
- Mit der `AllowManageOwnPasswords`-Anweisung kann der Benutzer sein eigenes Passwort ändern. Diese Anweisung enthält auch die `GetUser`-Aktion, die erforderlich ist, um die meisten Informationen auf der Seite My Security Credentials (Meine Sicherheitsanmeldeinformationen) anzuzeigen.
- Die `AllowManageOwnAccessKeys`-Anweisung ermöglicht dem Benutzer das Erstellen, Aktualisieren und Löschen seiner eigenen Zugriffsschlüssel. Der Benutzer kann auch Informationen darüber abrufen, wann der angegebene Zugriffsschlüssel zuletzt verwendet wurde.
- Die `AllowManageOwnSSHPublicKeys` Anweisung ermöglicht es dem Benutzer, seine eigenen öffentlichen SSH-Schlüssel für CodeCommit hochzuladen, zu aktualisieren und zu löschen.

Diese Richtlinie erlaubt es Benutzern nicht, ihre eigenen MFA-Geräte anzuzeigen oder zu verwalten. Außerdem können sie die Seite Users (Benutzer) in der IAM-Konsole nicht anzuzeigen oder für den Zugriff auf ihre eigenen Benutzerinformationen verwenden. Um dies zu ermöglichen, fügen Sie die Aktion `iam:ListUsers` zur Anweisung `AllowViewAccountInfo` hinzu. Darüber hinaus ermöglicht sie Benutzern nicht, ihr Passwort auf ihrer eigenen Benutzerseite anzupassen. Um dies zu ermöglichen, fügen Sie die Aktionen `iam:GetLoginProfile` und `iam:UpdateLoginProfile` zur Anweisung `AllowManageOwnPasswords` hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Verweigert den Zugriff auf AWS basierend auf der angeforderten Region

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den Zugriff auf alle Aktionen außerhalb der mit dem `aws:RequestedRegion`-Bedingungsschlüssel angegebenen Regionen verweigert, mit Ausnahme der Aktionen in den mit `NotAction` angegebenen Services. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den

Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Diese Richtlinie verwendet das `NotAction`-Element mit dem `Deny`-Effekt, der explizit den Zugriff auf alle Aktionen verweigert, die nicht in der Anweisung aufgeführt sind. Aktionen in den Diensten CloudFront, IAM, Route 53 und AWS Support Diensten sollten nicht verweigert werden, da es sich dabei um beliebte AWS globale Dienste mit einem einzigen Endpunkt handelt, der sich physisch in der `us-east-1` Region befindet. Da alle Anforderungen an diese Services an die Region `us-east-1` gerichtet werden, würden die Anforderungen ohne das `NotAction`-Element verweigert werden. Bearbeiten Sie dieses Element, um Aktionen für andere globale AWS -Services einzuschließen, die Sie verwenden, beispielsweise `budgets`, `globalaccelerator`, `importexport`, `organizations` oder `waf`. Einige andere globale Dienste, wie z. B. AWS Chatbot und AWS Device Farm, sind globale Dienste mit Endpunkten, die sich physisch in der `us-west-2` Region befinden. Weitere Informationen zu allen Services mit einem einzigen globalen Endpunkt finden Sie unter AWS -Regionen und Endpunkte in der Allgemeine AWS-Referenz. Weitere Informationen zur Verwendung des `NotAction`-Elements mit dem `Deny`-Effekt finden Sie unter IAM-JSON-Richtlinienelemente: NotAction.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die bestimmte Aktionen zulassen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

AWS: Verweigert den Zugriff auf AWS basierend auf der Quell-IP

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die den Zugriff auf alle AWS Aktionen im Konto verweigert, wenn die Anfrage von Principals außerhalb des angegebenen IP-Bereichs kommt. Die Richtlinie ist nützlich, wenn die IP-Adressen für Ihre Firma innerhalb der angegebenen Bereiche liegen. In diesem Beispiel wird die Anfrage abgelehnt, sofern sie nicht aus dem CIDR-Bereich 192.0.2.0/24 oder 203.0.113.0/24 stammt. Die Richtlinie lehnt Anfragen von AWS Diensten nicht ab, Forward Access Sessions (FAS) da die IP-Adresse des ursprünglichen Anforderers beibehalten wird.

Seien Sie vorsichtig mit negativen Bedingungen in der gleichen Richtlinienaussage wie `"Effect": "Deny"`. Wenn Sie dies tun, werden die in der Richtlinienanweisung angegebenen Aktionen unter allen Bedingungen explizit verweigert, mit Ausnahme der angegebenen.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die bestimmte Aktionen zulassen.

Wenn andere Richtlinien Aktionen zulassen, können Auftraggeber Anforderungen innerhalb des IP-Adressbereichs ausgeben. Ein AWS Dienst kann auch Anfragen mit den Anmeldeinformationen des Prinzipals stellen. Wenn ein Auftraggeber eine Anforderung außerhalb des IP-Bereichs ausgibt, wird die Anforderung abgelehnt.

Weitere Informationen über die Verwendung des Bedingungsschlüssels `aws:SourceIp`, einschließlich Informationen darüber, wann `aws:SourceIp` in Ihrer Richtlinie vielleicht nicht funktioniert, finden Sie unter AWS Kontextschlüssel für globale Bedingungen.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: Verweigern Sie den Zugriff auf Amazon S3 S3-Ressourcen außerhalb Ihres Kontos, außer AWS Data Exchange

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die den Zugriff auf alle Ressourcen verweigert AWS , die nicht zu Ihrem Konto gehören, mit Ausnahme der Ressourcen, die für den normalen Betrieb AWS Data Exchange erforderlich sind. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Sie können eine ähnliche Richtlinie erstellen, um den Zugriff auf Ressourcen innerhalb einer Organisation oder einer Organisationseinheit einzuschränken und gleichzeitig AWS Data Exchange eigene Ressourcen zu berücksichtigen, indem Sie die Bedingungsschlüssel und verwenden. `aws:ResourceOrgPaths` `aws:ResourceOrgID`

Wenn Sie den Service AWS Data Exchange in Ihrer Umgebung verwenden, erstellt er Ressourcen wie Amazon S3 S3-Buckets, die dem Servicekonto gehören, und interagiert mit diesen. AWS Data Exchange Sendet beispielsweise Anfragen an Amazon S3 S3-Buckets, die dem AWS Data Exchange Service gehören, im Namen des IAM-Prinzipals (Benutzer oder Rolle), der die APIs

aufruft. AWS Data Exchange In diesem Fall wird durch die Verwendung `aws:ResourceAccount` oder `aws:ResourceOrgID` in einer Richtlinie ohne Berücksichtigung AWS Data Exchange eigener Ressourcen der Zugriff auf die Buckets verweigert, die dem Servicekonto gehören. `aws:ResourceOrgPaths`

- Die Anweisung, `DenyAllAwsResourcesOutsideAccountExceptS3`, benutzt das `NotAction`-Element mit dem Deny-Effekt, der explizit den Zugriff auf jede Aktion verweigert, die nicht in der Anweisung aufgeführt ist und die auch nicht zum aufgelisteten Konto gehört. Das `NotAction`-Element gibt die Ausnahmen zu dieser Anweisung an. Diese Aktionen bilden die Ausnahme von dieser Aussage, denn wenn die Aktionen für Ressourcen ausgeführt werden, die von erstellt wurden AWS Data Exchange, werden sie von der Richtlinie verweigert.
- Die Anweisung, `DenyAllS3ResoucesOutsideAccountExceptDataExchange`, verwendet eine Kombination aus den `ResourceAccount`- und `CalledVia`-Bedingungen, um den Zugriff auf die drei Amazon-S3-Aktionen zu verweigern, die in der vorherigen Anweisung ausgeschlossen wurden. Die Anweisung lehnt die Aktionen ab, wenn Ressourcen nicht in das aufgelistete Konto gehören und wenn der aufrufende Service nicht AWS Data Exchange ist. Die Anweisung verweigert die Aktionen nicht, wenn entweder die Ressource dem aufgelisteten Konto gehört oder der aufgelistete Service-Prinzipal, `dataexchange.amazonaws.com`, die Vorgänge durchführt.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Sie verwendet den `Deny`-Effekt, der ausdrücklich den Zugriff auf alle in der Anweisung aufgeführten Ressourcen verweigert, die nicht zu dem aufgeführten Konto gehören. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die den Zugriff auf bestimmte Ressourcen gewähren.

Das folgende Beispiel zeigt, wie Sie die Richtlinie konfigurieren können, um den Zugriff auf die erforderlichen Amazon-S3-Buckets zu ermöglichen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
```

```
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Data Pipeline: Verweigert den Zugriff auf DataPipeline Pipelines, die ein Benutzer nicht erstellt hat

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den Zugriff auf Pipelines verweigert, die ein Benutzer nicht erstellt hat. Wenn der Wert des `PipelineCreator`-Feldes dem IAM-Benutzernamen entspricht, dann werden die angegebenen Aktionen nicht

verweigert. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die API oder durchzuführen. AWS AWS CLI

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die bestimmte Aktionen zulassen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator": "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB: Gewährt Zugriff auf eine bestimmte Tabelle

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die den vollen Zugriff auf die `MyTable`-DynamoDB-Tabelle erlaubt. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

> **Important**
>
> Diese Richtlinie lässt alle Aktionen zu, die für eine DynamoDB-Tabelle ausgeführt werden können. Eine Übersicht über diese Aktionen finden Sie unter DynamoDB-API-Berechtigungen: Aktionen Ressourcen und Bedingungen Leitfaden im Amazon DynamoDB Developer Guide. Sie könnten dieselben Berechtigungen bereitstellen, indem Sie jede einzelne Aktion aufführen. Wenn Sie jedoch den Platzhalter (`*`) im `Action`-Element verwenden, z. B. `"dynamodb:List*"`, dann müssen Sie Ihre Richtlinie nicht aktualisieren, wenn DynamoDB eine neue List-Aktion hinzufügt.

Diese Richtlinie lässt Aktionen nur für die DynamoDB-Tabellen zu, die mit dem angegebenen Namen vorhanden sind. Um Ihren Benutzern `Read` Zugriff auf alles in DynamoDB zu gewähren, können Sie auch die AmazonDynamoReadOnlyAccess AWS DB-verwaltete Richtlinie anhängen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
```

```
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB: Ermöglicht den Zugriff auf bestimmte Attribute

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den Zugriff auf die spezifischen DynamoDB-Attribute erlaubt. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Die Anforderung `dynamodb:Select` verhindert, dass die API-Aktion jegliche Attribute, die nicht zulässig sind, zurückgibt, z. B. von einer Indexprojektion. Weitere Informationen zu DynamoDB-Bedingungsschlüsseln finden Sie unter Specifying Conditions: Using Condition Keys im Amazon DynamoDB Developer Guide. Weitere Informationen zum Verwenden mehrerer Bedingungen oder mehrerer Bedingungsschlüssel innerhalb eines `Condition`-Blocks einer IAM-Richtlinie finden Sie unter Mehrere Werte in einer Bedingung.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select": "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB: Ermöglicht den Zugriff auf DynamoDB auf Elementebene basierend auf einer Amazon Cognito ID

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den Zugriff auf Elementebene auf die `MyTable`-DynamoDB-Tabelle anhand einer Amazon-Cognito-Identitätspool-ID ermöglicht. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Um diese Richtlinie nutzen zu können, müssen Sie die DynamoDB-Tabelle so strukturieren, dass die Identitätspool-Benutzer-ID für Amazon Cognito der Partitionsschlüssel ist. Weitere Informationen finden Sie unter Erstellen eeiner Tabelle im Amazon DynamoDB Developer Guide.

Weitere Informationen zu DynamoDB-Bedingungsschlüsseln finden Sie unter Specifying Conditions: Using Condition Keys im Amazon DynamoDB Developer Guide.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2: Anfügen von Amazon EBS-Volumes anhand von Tags an EC2-Instances oder Trennen dieser Volumes

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es Besitzern von EBS-Volumes erlaubt, ihre mit dem Tag `VolumeUser` definierten EBS-Volumes an EC2-Instances anzufügen oder zu entfernen, die als Entwicklungs-Instances (`Department=Development`) gekennzeichnet sind. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Weitere Informationen zur Erstellung von IAM-Richtlinien zur Steuerung des Zugriffs auf Amazon EC2 EC2-Ressourcen finden Sie unter Steuern des Zugriffs auf Amazon EC2 EC2-Ressourcen im Amazon EC2 EC2-Benutzerhandbuch.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2: Ermöglicht es, EC2-Instances in einem bestimmten Subnetz programmgesteuert und in der Konsole zu starten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Auflisten von Informationen für alle EC2-Objekte und das Launchen von EC2-Instances in einem bestimmten Subnetz erlaubt. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
```

```
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2: Ermöglicht es, die mit einem bestimmten Tag-Schlüsselwertpaar verknüpfte EC2-Sicherheitsgruppen programmgesteuert und in der Konsole zu verwalten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die Benutzern die Berechtigung erteilt, bestimmte Aktionen für Sicherheitsgruppen mit demselben Tag durchzuführen. Die folgende Richtlinie erteilt die Berechtigung, Sicherheitsgruppen in der Amazon-EC2-Konsole anzuzeigen, Regeln für ein- und ausgehenden Datenverkehr hinzuzufügen und zu entfernen und Regelbeschreibungen für vorhandene Sicherheitsgruppen, die über das `Department=Test-` Tag verfügen, aufzulisten und zu ändern. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
```

```
            "ec2:DescribeSecurityGroupRules",
            "ec2:DescribeTags"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:AuthorizeSecurityGroupIngress",
            "ec2:RevokeSecurityGroupIngress",
            "ec2:AuthorizeSecurityGroupEgress",
            "ec2:RevokeSecurityGroupEgress",
            "ec2:ModifySecurityGroupRules",
            "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
            "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
        ],
        "Resource": [
            "arn:aws:ec2:region:111122223333:security-group/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/Department": "Test"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:ModifySecurityGroupRules"
        ],
        "Resource": [
            "arn:aws:ec2:region:111122223333:security-group-rule/*"
        ]
    }
  ]
}
```

Amazon EC2: Ermöglicht es, die von einem Benutzer markierten EC2-Instances programmgesteuert und in der Konsole zu starten oder zu stoppen

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es einem IAM-Benutzer erlaubt, EC2-Instances zu starten oder zu stoppen, jedoch nur wenn das Instance-

Tag `Owner` den Wert des Benutzernamens dieses Benutzers hat. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2: Starten oder Stoppen von Instances basierend auf Tags

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Starten oder Stoppen von Instances mit dem Tag-Schlüssel-Wert-Paar `Project = DataAnalytics` erlaubt, aber nur durch Prinzipale mit dem Tag-Schlüssel-Wert-Paar `Department = Data`. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Die Bedingung in der Richtlinie gibt "true" zurück, wenn beide Teile der Bedingung erfüllt sind. Die Instance muss das `Project=DataAnalytics`-Tag aufweisen. Darüber hinaus muss der IAM-Auftraggeber (Benutzer oder Rolle), von dem die Anforderung stammt, über das `Department=Data`-Tag verfügen.

> **Note**
>
> Als bewährte Methode können Sie Richtlinien mit dem `aws:PrincipalTag`-Bedingungsschlüssel an IAM-Gruppen anfügen, für den Fall, dass einige Benutzer das angegebene Tag haben und andere nicht.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2: Starten oder Stoppen von Instances basierend auf übereinstimmenden Auftraggeber- und Ressourcen-Tags

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es einem Prinzipal erlaubt, eine Amazon-EC2-Instance zu starten oder zu stoppen, wenn das Ressourcen-Tag der Instance und das Tag des Prinzipals denselben Wert für den Tag-Schlüssel `CostCenter` haben. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

> **Note**
>
> Als bewährte Methode können Sie Richtlinien mit dem `aws:PrincipalTag`-Bedingungsschlüssel an IAM-Gruppen anfügen, für den Fall, dass einige Benutzer das angegebene Tag haben und andere nicht.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2: Gewährt EC2-Vollzugriff innerhalb einer bestimmten Region programmatisch und in der Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die vollen EC2-Zugriff innerhalb einer bestimmten Region erlaubt. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie. Eine Liste der Regionscodes finden Sie unter Verfügbare Regionen im Amazon EC2 Leitfaden.

Alternativ können Sie den globalen Bedingungsschlüssel `aws:RequestedRegion` verwenden, der von allem Amazon EC2-API-Aktionen unterstützt wird. Weitere Informationen finden Sie unter Beispiel: Einschränken des Zugriffs auf eine bestimmte Region im Amazon EC2 Leitfaden.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2: Ermöglicht das Starten oder Beenden einer EC2-Instance und das Ändern einer Sicherheitsgruppe (programmatisch und in der Konsole)

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Starten oder Beenden einer bestimmten EC2-Instance und das Ändern einer bestimmten Sicherheitsgruppe erlaubt. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
```

```
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Amazon EC2: Erfordert MFA (GetSessionToken) für bestimmte EC2-Operationen

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die vollen Zugriff auf alle AWS API-Operationen in Amazon EC2 ermöglicht. Dabei wird jedoch explizit der Zugriff auf die API-Operationen `StopInstances` und `TerminateInstances` verweigert, wenn der Benutzer nicht über die Multi-Factor Authentication (MFA) authentifiziert wurde. Um dies programmgesteuert durchzuführen, muss der Benutzer optionale `SerialNumber`- und `TokenCode`-Werte beim Aufruf der `GetSessionToken`-Operation einschließen. Diese Operation gibt temporäre Anmeldeinformationen zurück, die per MFA authentifiziert wurden. Weitere Informationen GetSessionToken dazu finden Sie unter. GetSessionToken - vorläufige Anmeldedaten für Benutzer in nicht vertrauenswürdigen Umgebungen

Was macht diese Richtlinie?

- Durch die `AllowAllActionsForEC2`-Anweisung werden alle Amazon EC2-Aktionen gewährt.
- Die `DenyStopAndTerminateWhenMFAIsNotPresent`-Anweisung verweigert die Aktionen `TerminateInstances` und `StopInstances`, wenn der MFA-Kontext fehlt. Das bedeutet, dass die Aktionen verweigert werden, wenn der MFA-Kontext fehlt (also keine MFA stattgefunden hat). Eine Verweigerung überschreibt mögliche Berechtigungen.

> **Note**
>
> Für die Bedingungsprüfung für `MultiFactorAuthPresent` in der `Deny`-Anwendung sollte nicht `{"Bool":{"aws:MultiFactorAuthPresent":false}}` verwendet werden, da der Schlüssel nicht vorhanden ist und auch nicht ausgewertet werden kann, wenn keine MFA verwendet wird. Verwenden Sie stattdessen die Prüfung `BoolIfExists`, um zu sehen, ob

der Schlüssel vorhanden ist, bevor Sie den Wert prüfen. Weitere Informationen finden Sie unter ... IfExists Bedingungsoperatoren.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2: Beschränkt das Beenden von EC2-Instances auf einen IP-Adressbereich

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die EC2-Instances einschränkt, indem sie die Aktion zulässt, aber den Zugriff ausdrücklich verweigert, wenn die Anforderung von außerhalb des angegebenen IP-Bereichs kommt. Die Richtlinie ist nützlich, wenn die IP-Adressen für Ihre Firma innerhalb der angegebenen Bereiche liegen. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Wenn diese Richtlinie in Kombination mit anderen Richtlinien verwendet wird, die die `ec2:TerminateInstances` Aktion zulassen (z. B. der von AmazonEC2 FullAccess AWS verwalteten Richtlinie), wird der Zugriff verweigert. Der Grund hierfür liegt darin, dass eine explizite Zugriffsverweigerung Vorrang vor einer Zugriffserlaubnis hat. Weitere Informationen finden Sie unter the section called “Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird”.

> **Important**
>
> Der `aws:SourceIp` Bedingungsschlüssel verweigert den Zugriff auf einen AWS Dienst, der beispielsweise in AWS CloudFormation Ihrem Namen Anrufe tätigt. Weitere Informationen zum Verwenden des `aws:SourceIp`-Bedingungsschlüssels finden Sie unter AWS Kontextschlüssel für globale Bedingungen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM: Zugriff auf die Richtliniensimulator-API

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Verwendung der Richtliniensimulator-API für Richtlinien erlaubt, die an einen Benutzer, eine Gruppe oder eine Rolle im aktuellen AWS-Konto angefügt sind. Diese Richtlinie ermöglicht auch den Zugriff auf weniger sensible Richtlinien, die als Zeichenfolgen an die API übergeben werden. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Informationen darüber, wie ein Benutzer auf die Richtliniensimulator-Konsole zugreifen kann, um Richtlinien zu simulieren, die einem Benutzer, einer Gruppe oder einer Rolle in der aktuellen Version zugewiesen sind AWS-Konto, finden Sie unter. IAM: Zugriff auf die Richtliniensimulator-Konsole

IAM: Zugriff auf die Richtliniensimulator-Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Verwendung der Richtliniensimulator-Konsole für Richtlinien erlaubt, die an einen Benutzer, einer Gruppe oder einer Rolle im aktuellen AWS-Konto angefügt sind. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI

Sie erhalten Zugriff auf die IAM-Richtliniensimulator-Konsole unter: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM: Übernehmen von Rollen, die ein bestimmtes Tag haben

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es einem IAM-Benutzer erlaubt, Rollen mit dem Tag-Schlüssel-Wert-Paar `Project = ExampleCorpABC` anzunehmen. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Wenn eine Rolle mit diesem Tag im selben Konto wie der Benutzer vorhanden ist, kann der Benutzer diese Rolle übernehmen. Wenn eine Rolle mit diesem Tag in einem anderen Konto als dem Konto des Benutzers vorhanden ist, sind zusätzliche Berechtigungen erforderlich. Die Vertrauensrichtlinie der kontoübergreifenden Rolle muss dem Benutzer oder allen Mitgliedern des Benutzerkontos erlauben, diese Rolle zu übernehmen. Weitere Informationen zum Verwenden von Rollen für den kontoübergreifenden Zugriff finden Sie unter Bereitstellen des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM: Ermöglicht und verweigert den Zugriff auf verschiedene Services, sowohl programmgesteuert als auch über die Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die einen vollständigen Zugriff auf verschiedene Services und begrenzten selbstverwaltenden Zugriff in IAM erlaubt. Verweigert Benutzern den Zugriff auf den Amazon S3 Protokoll-Bucket `logs` oder die `i-1234567890abcdef0`-Amazon EC2-Instance. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

> **Warning**
>
> Diese Richtlinie ermöglicht den vollständigen Zugriff auf alle Aktionen und Ressourcen in verschiedenen Services. Diese Richtlinie sollte nur auf vertrauenswürdige Administratoren angewendet werden.

Sie können diese Richtlinien als Berechtigungsgrenze einsetzen, um die maximale Anzahl von Berechtigungen festzulegen, die eine identitätsbasierte Richtlinie einem IAM-Benutzer erteilen kann. Weitere Informationen finden Sie unter Verantwortung mit Hilfe von Berechtigungsgrenzen an andere delegieren. Wenn die Richtlinie als Berechtigungsgrenze für einen Benutzer verwendet wird, wird durch die Anweisungen die folgende Grenze festgelegt:

- Die `AllowServices`-Anweisung gewährt den angegebenen AWS -Services einen Vollzugriff. Dies bedeutet, dass die Aktionen eines Benutzers in diesen Services nur laut der Berechtigungsrichtlinien begrenzt sind, die dem Benutzer zugeordnet sind.
- Die `AllowIAMConsoleForCredentials`-Anweisung erlaubt den Zugriff für das Auflisten aller IAM-Benutzer. Dieser Zugriff ist erforderlich, um auf der Seite Users (Benutzer) in der AWS Management Console navigieren zu können. Sie ermöglicht zudem das Anzeigen von Passwortanforderungen für das Konto. Dies ist erforderlich, damit der Benutzer sein eigenes Passwort ändern kann.
- Mit der `AllowManageOwnPasswordAndAccessKeys`-Anweisung können die Benutzer nur ihr eigenes Konsolen-Passwort und programmatische Zugriffsschlüssel verwalten. Dies ist wichtig, denn wenn eine andere Richtlinie einem Benutzer einen vollständigen IAM-Zugriff gewährt, kann dieser Benutzer seine eigenen Berechtigungen oder die eines anderen Benutzers ändern. Diese Anweisung verhindert, dass dies passiert.
- Die Anweisung `DenyS3Logs` verweigert explizit den Zugriff auf den `logs`-Bucket. Diese Richtlinie sorgt dafür, dass einem Benutzer Unternehmensbeschränkungen auferlegt werden.
- Die Anweisung `DenyEC2Production` verweigert explizit den Zugriff auf die `i-1234567890abcdef0`-Instance.

Diese Richtlinie gewährt nicht den Zugriff auf andere Services oder Aktionen. Wenn die Richtlinie als Berechtigungsgrenze für einen Benutzer verwendet wird, wird die Anfrage AWS abgelehnt, auch wenn andere dem Benutzer zugeordnete Richtlinien diese Aktionen zulassen.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
```

```
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM: Hinzufügen eines bestimmten Tags zu einem Benutzer mit einem bestimmten Tag

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Hinzufügen des Tag-Schlüssels `Department` mit den Tag-Werten `Marketing`, `Development` oder `QualityAssurance` zu einem IAM-Benutzer erlaubt. Dieser Benutzer muss bereits das Tag-Schlüssel-Wert-Paar `JobFunction = manager` enthalten. Mit dieser Richtlinie können Sie festlegen, dass ein Manager nur zu einer von drei Abteilungen gehören kann. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Mit der `ListTagsForAllUsers`-Anweisung kann die Anzeige von Tags für alle Benutzer in Ihrem Konto erlaubt werden.

Die erste Bedingung in der `TagManagerWithSpecificDepartment`-Anweisung verwendet den `StringEquals`-Bedingungsoperator. Die Bedingung gibt "true" zurück, wenn beide Teile der Bedingung erfüllt sind. Der zu markierende Benutzer muss bereits über das `JobFunction=Manager`-Tag verfügen. Die Anforderung muss den `Department`-Tag-Schlüssel mit einem der aufgelisteten Tag-Werte aufweisen.

Die zweite Bedingung verwendet den `ForAllValues:StringEquals`-Bedingungsoperator. Die Bedingung gibt "true" zurück, wenn alle Tag-Schlüssel in der Anforderung mit dem Schlüssel in der Richtlinie übereinstimmen. Das bedeutet, dass der einzige Tag-Schlüssel in der Anforderung `Department` sein muss. Weitere Informationen zur Verwendung von `ForAllValues` finden Sie unter Mehrwertige Kontextschlüssel.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM: Hinzufügen eines bestimmten Tags mit bestimmten Werten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die nur das Hinzufügen des Tag-Schlüssels `CostCenter` und entweder des Tag-Werts `A-123` oder des Tag-Werts `B-456` zu einem beliebigen IAM-Benutzer oder einer Rolle erlaubt. Sie können diese Richtlinie verwenden, um das Markieren auf einen bestimmten Tag-Schlüssel und eine Reihe von Tag-Werten zu beschränken. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Mit der `ConsoleDisplay`-Anweisung kann die Anzeige von Tags für alle Benutzer und Rollen in Ihrem Konto erlaubt werden.

Die erste Bedingung in der `AddTag`-Anweisung verwendet den `StringEquals`-Bedingungsoperator. Die Bedingung gibt "true" zurück, wenn die Anforderung den `CostCenter`-Tag-Schlüssel mit einem der aufgelisteten Tag-Werte aufweist.

Die zweite Bedingung verwendet den `ForAllValues:StringEquals`-Bedingungsoperator. Die Bedingung gibt "true" zurück, wenn alle Tag-Schlüssel in der Anforderung mit dem Schlüssel in der Richtlinie übereinstimmen. Das bedeutet, dass der einzige Tag-Schlüssel in der Anforderung `CostCenter` sein muss. Weitere Informationen zur Verwendung von `ForAllValues` finden Sie unter Mehrwertige Kontextschlüssel.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM: Erstellen neuer Benutzer nur mit bestimmten Tags

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Erstellung von IAM-Benutzern erlaubt, aber nur mit einem oder beiden Tag-Schlüsseln `Department` und `JobFunction`. Der `Department`-Tag-Schlüssel muss entweder den `Development`- oder den `QualityAssurance`-Tag-Wert aufweisen. Der `JobFunction`-Tag-Schlüssel muss den `Employee`-Tag-Wert aufweisen. Mit dieser Richtlinie können Sie festlegen, dass neue Benutzer eine bestimmte Stellenfunktion und Abteilung aufweisen müssen. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Die erste Bedingung in der Anweisung verwendet den `StringEqualsIfExists`-Bedingungsoperator. Wenn ein Tag mit dem `Department`- oder dem `JobFunction`-Schlüssel in der Anforderung vorhanden ist, muss das Tag den angegebenen Wert aufweisen. Wenn kein Schlüssel vorhanden ist, wird diese Bedingung als "true" ausgewertet. Die einzige Möglichkeit, dass die Bedingung als "false" ausgewertet wird, ist gegeben, wenn einer der angegebenen Bedingungsschlüssel in der Anforderung vorhanden ist, aber einen anderen Wert als die zulässigen hat. Weitere Informationen zur Verwendung von `IfExists` finden Sie unter ... IfExists Bedingungsoperatoren.

Die zweite Bedingung verwendet den `ForAllValues:StringEquals`-Bedingungsoperator. Die Bedingung gibt "true" zurück, wenn alle in der Anforderung angegebenen Tag-Schlüssel mit mindestens einem Richtlinienwert übereinstimmen. Das bedeutet, dass alle Tags in der Anforderung in dieser Liste aufgeführt sein müssen. Die Anforderung kann jedoch nur einen der Tags in der Liste enthalten. Sie können beispielsweise einen IAM-Benutzer nur mit dem `Department=QualityAssurance`-Tag erstellen. Es ist jedoch nicht möglich, einen IAM-Benutzer mit dem `JobFunction=employee`-Tag und dem `Project=core`-Tag zu erstellen. Weitere Informationen zur Verwendung von `ForAllValues` finden Sie unter Mehrwertige Kontextschlüssel.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
```

```
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM: Generieren und Abrufen von -Berichten zu Anmeldeinformationen

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es Benutzern ermöglicht, einen Bericht zu generieren und herunterzuladen, in dem alle IAM-Benutzer in ihrem Verzeichnis aufgeführt sind. AWS-Konto Der Bericht enthält außerdem den Status der Anmeldeinformationen des Benutzers, einschließlich Passwörter, MFA-Geräte und Signaturzertifikate. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die API oder durchzuführen. AWS AWS CLI

Weitere Informationen zu Berichten zu Anmeldeinformationen finden Sie unter Abrufen von Berichten zu Anmeldeinformationen für Ihr AWS-Konto.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:GetCredentialReport"
        ],
```

```
        "Resource": "*"
    }
}
```

IAM: Ermöglicht das Verwalten der Mitgliedschaft einer Gruppe sowohl programmgesteuert als auch über die Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Aktualisierung der Mitgliedschaft der Gruppe mit dem Namen `MarketingTeam` erlaubt. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Was macht diese Richtlinie?

- Die `ViewGroups`-Anweisung erlaubt dem Benutzer das Auflisten aller Benutzer und Gruppen in der AWS Management Console. Darüber hinaus ermöglicht sie dem Benutzer das Anzeigen grundlegender Informationen zu den Benutzern im Konto. Diese Berechtigungen müssen in ihrer eigenen Anweisung vorliegen, da sie keine Ressourcen-ARN unterstützen oder keine angeben müssen. Geben Sie anstelle von Berechtigungen `"Resource" : "*"` ein.
- Die `ViewEditThisGroup`-Anweisung ermöglicht dem Benutzer, Informationen zur Gruppe `MarketingTeam` anzuzeigen und Benutzer zu dieser Gruppe hinzuzufügen oder aus dieser zu entfernen.

Diese Richtlinie lässt nicht zu, dass der Benutzer die Berechtigungen der Benutzer oder der Gruppe `MarketingTeam` anzeigt oder bearbeitet.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM: Verwalten eines bestimmten Tags

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die das Hinzufügen und Entfernen des IAM-Tags mit dem Tag-Schlüssel `Department` von IAM-Entitäten (Benutzern und Rollen) erlaubt. Diese Richtlinie schränkt den Wert des `Department`-Tags nicht ein. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM: Übergeben Sie eine IAM-Rolle an einen bestimmten Dienst AWS

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die es ermöglicht, jede IAM-Servicerolle an den Amazon-Service zu übergeben. CloudWatch Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die API oder durchzuführen. AWS AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Eine Servicerolle ist eine IAM-Rolle, die einen AWS Dienst als Principal angibt, der die Rolle übernehmen kann. Auf diese Weise kann der Service die Rolle übernehmen und in Ihrem Namen auf Ressourcen eines anderen Services zugreifen. Damit Amazon CloudWatch die Rolle übernehmen kann, die Sie bestehen, müssen Sie in der Vertrauensrichtlinie Ihrer Rolle den `cloudwatch.amazonaws.com` Service Principal als Principal angeben. Der Dienstauftraggeber wird durch den Service definiert. Um den Dienstauftraggeber für einen Service zu ermitteln, lesen Sie die Dokumentation für diesen Service. Für einige Services finden Sie Informationen unter AWS Dienste, die mit IAM funktionieren und suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen. Suchen Sie nach `amazonaws.com`, um den Dienstauftraggeber anzuzeigen.

Weitere Informationen zum Übergeben einer Servicerolle an einen Service finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService": "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM: Ermöglicht einen schreibgeschützten Zugriff auf die IAM-Konsole ohne Berichterstellung

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die IAM-Benutzern die Berechtigung zum Ausführen einer IAM-Aktion gewährt, die mit der Zeichenfolge `Get` oder `List` beginnt. Wenn Benutzer die Konsole verwenden, stellt diese Anforderungen an IAM, um Gruppen, Benutzer, Rollen und Richtlinien aufzulisten und Berichte über diese Ressourcen zu generieren.

Das Sternchen dient als Platzhalter. Wenn Sie `iam:Get*` in einer Richtlinie verwenden, enthalten die resultierenden Berechtigungen alle IAM-Aktionen, die mit `Get` beginnen, beispielsweise `GetUser` und `GetRole`. Platzhalter sind nützlich, wenn neue Entitätstypen zu IAM hinzugefügt werden. In diesem Fall wird es dem Benutzer durch die Berechtigungen, die laut der Richtlinie gewährt werden, automatisch erlaubt, Details zu diesen neuen Entitäten aufzulisten und abzurufen.

Diese Richtlinie kann nicht zum Generieren von Berichten oder Dienstdetails verwendet werden, auf die zuletzt zugegriffen wird. Eine andere Richtlinie, die dies zulässt, finden Sie unter IAM: Gewährt einen schreibgeschützten Zugriff auf die IAM-Konsole.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

IAM: Gewährt einen schreibgeschützten Zugriff auf die IAM-Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die IAM-Benutzern die Berechtigung zum Ausführen einer IAM-Aktion gewährt, die mit der Zeichenfolge `Get`, `List` oder `Generate` beginnt. Wenn Benutzer die IAM-Konsole verwenden, stellt diese Anforderungen, um Gruppen, Benutzer, Rollen und Richtlinien aufzulisten und Berichte über diese Ressourcen zu generieren.

Das Sternchen dient als Platzhalter. Wenn Sie `iam:Get*` in einer Richtlinie verwenden, enthalten die resultierenden Berechtigungen alle IAM-Aktionen, die mit `Get` beginnen, beispielsweise

`GetUser` und `GetRole`. Das Verwenden eines Platzhalters ist hilfreich, insbesondere, wenn neue Entitätstypen zu IAM hinzugefügt werden. In diesem Fall wird es dem Benutzer durch die Berechtigungen, die laut der Richtlinie gewährt werden, automatisch erlaubt, Details zu diesen neuen Entitäten aufzulisten und abzurufen.

Verwenden Sie diese Richtlinie für den Konsolenzugriff, der Berechtigungen zum Generieren von Berichten oder Dienstdetails enthält. Für eine andere Richtlinie, die das Erzeugen von Aktionen nicht zulässt, siehe IAM: Ermöglicht einen schreibgeschützten Zugriff auf die IAM-Konsole ohne Berichterstellung.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*",
            "iam:Generate*"
        ],
        "Resource": "*"
    }
}
```

: Ermöglicht spezifischen IAM-Benutzern das Verwalten einer Gruppe programmgesteuert und in der Konsole

Dieses Beispiel zeigt, wie Sie eine IAM-Richtlinie erstellen könnten, die es bestimmten IAM-Benutzern erlaubt, die `AllUsers`-Gruppe zu verwalten. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Was macht diese Richtlinie?

- Die `AllowAllUsersToListAllGroups`-Anweisung erlaubt das Auflisten aller Gruppen. Dies ist für den Konsolenzugriff erforderlich. Diese Berechtigung muss als eigene Anweisung vorliegen, da sie keinen Ressourcen-ARN unterstützt. Geben Sie anstelle von Berechtigungen `"Resource" : "*"` ein.

- Die `AllowAllUsersToViewAndManageThisGroup`-Anweisung lässt alle Gruppenaktionen zu, die für den Gruppenressourcentyp durchgeführt werden können. Sie lässt die `ListGroupsForUser`-Aktion nicht zu. Diese Aktion kann für einen Benutzerressourcentyp und nicht für einen Gruppenressourcentyp durchgeführt werden. Weitere Informationen über die Ressourcentypen, die Sie für eine IAM-Aktion angeben können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Identity and Access Management.
- Die `LimitGroupManagementAccessToSpecificUsers`-Anweisung verweigert Benutzern mit den angegebenen Namen den Zugriff auf Gruppenaktionen für die Schreib- und Berechtigungsverwaltung. Wenn ein in der Richtlinie angegebener Benutzer versucht, Änderungen an der Gruppe vorzunehmen, lehnt diese Anweisung die Anforderung nicht ab. Diese Anforderung wird durch die `AllowAllUsersToViewAndManageThisGroup`-Anweisung zugelassen. Wenn andere Benutzer versuchen, diese Operationen durchzuführen, wird die Anforderung abgelehnt. Sie können die -Aktionen, die über die Zugriffsebenen Write (Schreiben) oder Permissions management (Berechtigungsverwaltung) definiert wurden, anzeigen, während Sie die Richtlinie in der IAM-Konsole erstellen. Wechseln Sie dazu von der Registerkarte JSON zur RegisterkarteVisual editor (Visueller Editor) . Weitere Informationen über Zugriffsebenen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Identity and Access Management.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
```

```
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
                }
            }
        }
    ]
}
```

IAM: Ermöglicht das Festlegen der Anforderungen an das Kontopasswort programmgesteuert und in der Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es einem Benutzer erlaubt, die Passwort-Anforderungen für sein Konto anzuzeigen und zu aktualisieren. Die Passwortanforderungen umfassen die Anforderungen an die Komplexität sowie die vorgeschriebenen Wechselperioden für die Passwörter der Kontomitglieder. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

Informationen zum Festlegen der Anforderungen an das Kontopasswort für Ihr Konto finden Sie unter Einrichten einer Kontopasswortrichtlinie für IAM-Benutzer.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:UpdateAccountPasswordPolicy"
        ],
```

```
        "Resource": "*"
    }
}
```

IAM: Zugriff auf die Richtliniensimulator-API basierend auf dem Benutzerpfad

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Verwendung der Richtliniensimulator-API nur für Benutzer mit dem Pfad `Department/Development` zulässt. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> Informationen zum Erstellen einer Richtlinie, die die Verwendung der Richtliniensimulator-Konsole für diejenigen Benutzer erlaubt, die über den Pfad `Department/Development` verfügen, finden Sie unter IAM: Zugriff auf die Richtliniensimulator-Konsole basierend auf dem Benutzerpfad.

IAM: Zugriff auf die Richtliniensimulator-Konsole basierend auf dem Benutzerpfad

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Verwendung der Richtliniensimulator-Konsole nur für die Benutzer mit dem Pfad `Department/`

`Development` zulässt. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Sie erhalten Zugriff auf den IAM-Richtliniensimulator unter: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

IAM: Ermöglicht es IAM-Benutzern, MFA-Geräte selbst zu verwalten

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die IAM-Benutzern erlaubt, ihr Multi-Faktor-Authentifizierung (MFA)-Gerät selbst zu verwalten. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI

> **Note**
>
> Wenn ein IAM-Benutzer mit dieser Richtlinie nicht MFA-authentifiziert ist, verweigert diese Richtlinie den Zugriff auf alle AWS Aktionen außer denen, die für die Authentifizierung mit MFA erforderlich sind. Wenn Sie diese Berechtigungen für einen Benutzer hinzufügen, bei dem er angemeldet ist, muss er sich möglicherweise ab- und wieder anmelden AWS, um die Änderungen zu sehen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

IAM: Ermöglicht es IAM-Benutzern, ihre eigenen Anmeldeinformationen programmgesteuert und in der Konsole zu aktualisieren

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es IAM-Benutzern ermöglicht, ihre eigenen Zugriffsschlüssel, Signierzertifikate, servicespezifischen Anmeldeinformationen und Passwörter zu aktualisieren. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Weitere Informationen dazu, wie ein Benutzer sein eigenes Passwort in der Konsole ändern kann, finden Sie unter the section called “Wie ein IAM-Benutzer sein eigenes Passwort ändert”.

IAM: Anzeigen von Informationen zum letzten Zugriff auf einen Service für eine Organizations-Richtlinie

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Anzeige von Informationen über den letzten Zugriff auf einen Service für eine bestimmte Organisationsrichtlinie erlaubt. Diese Richtlinie ermöglicht das Abrufen von Daten für die Service-Kontrollrichtlinie (Service Control Policy, SCP) mit der `p-policy123`-ID. Die Person, die den Bericht generiert und anzeigt, muss mit den Anmeldeinformationen für das AWS Organizations Verwaltungskonto authentifiziert werden. Diese Richtlinie ermöglicht dem Anforderer, die Daten für alle Organizations-Entitäten in ihrer Organisation abzurufen. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Weitere Informationen zu Informationen zum letzten Servicezugriff, darunter zu den erforderlichen Berechtigungen und den unterstützten Regionen, finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
            }
        }
    ]
}
```

IAM: Beschränkt die verwalteten Richtlinien, die -Benutzern, -Gruppen oder -Rollen zugeordnet werden können

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die vom Kunden verwaltete und AWS verwaltete Richtlinien einschränkt, die auf einen IAM-Benutzer, eine Gruppe oder eine IAM-Rolle angewendet werden können. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die API oder durchzuführen.
AWS AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS: Verweigern Sie den Zugriff auf Ressourcen außerhalb Ihres Kontos mit Ausnahme von AWS verwalteten IAM-Richtlinien

Die Verwendung von `aws:ResourceAccount` in Ihren identitätsbasierten Richtlinien kann sich auf die Fähigkeit des Benutzers oder der Rolle auswirken, einige Services zu nutzen, die eine Interaktion mit Ressourcen in Konten erfordern, die einem Service gehören.

Sie können eine Richtlinie mit einer Ausnahme erstellen, um AWS verwaltete IAM-Richtlinien zuzulassen. Ein vom Service verwaltetes Konto außerhalb Ihrer AWS Organizations besitzt verwaltete IAM-Richtlinien. Es gibt vier IAM-Aktionen, die verwaltete Richtlinien auflisten und abrufen AWS. Verwenden Sie diese Aktionen im `NotAction`-Element der Anweisung. `AllowAccessToS3ResourcesInSpecificAccountsAndSpecificService1` in der Richtlinie.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
```

```
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda: Ermöglicht einer Lambda-Funktion den Zugriff auf eine Amazon DynamoDB -Tabelle

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den Lese- und Schreibzugriff auf eine bestimmte Amazon-DynamoDB-Tabelle erlaubt. Die Richtlinie ermöglicht auch das Schreiben von Protokolldateien in CloudWatch Logs. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Zur Nutzung dieser Richtlinie fügen Sie sie an eine Lambda Servicerolle an. Eine Servicerolle ist eine Rolle, die Sie in Ihrem Konto erstellen, um einem Service das Ausführen von Aktionen in Ihrem Namen zu ermöglichen. Diese Servicerolle muss AWS Lambda als Principal in der Vertrauensrichtlinie enthalten sein. Einzelheiten zur Verwendung dieser Richtlinie finden Sie unter So erstellen Sie eine AWS IAM-Richtlinie, um AWS Lambda Zugriff auf eine Amazon DynamoDB-Tabelle zu gewähren im AWS Sicherheitsblog.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
```

```
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: Gewährt Vollzugriff auf die RDS-Datenbank innerhalb einer bestimmten Region

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die den vollständigen RDS-Datenbankzugriff innerhalb einer bestimmten Region ermöglicht. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:region:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS: Ermöglicht es, RDS-Datenbanken programmgesteuert und in der Konsole wiederherzustellen

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Wiederherstellung von RDS-Datenbanken erlaubt. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Amazon RDS: Gewährt Tag-Eigentümern Vollzugriff auf die RDS-Ressourcen, die sie markiert haben

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die Tag-Besitzern vollen Zugriff auf RDS-Ressourcen gewährt, die sie markiert haben. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
```

```
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3: Ermöglicht Amazon Cognito-Benutzern den Zugriff auf Objekte in ihrem Bucket

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die Amazon-Cognito-Benutzern den Zugriff auf Objekte in einem bestimmten S3-Bucket ermöglicht. Diese Richtlinie gewährt den Zugriff nur auf Objekte mit einem Namen, der `cognito` enthält, den Namen der Anwendung und die ID des Verbundbenutzers enthält, dargestellt durch die Variable ${cognito-identity.:sub} variable. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

> **Note**
>
> Der im Objektschlüssel verwendete "sub"-Wert ist nicht der Sub-Wert des Benutzers im Benutzerpool, sondern die Identitäts-ID, die dem Benutzer im Identitätspool zugeordnet ist.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito bietet Authentifizierung, Autorisierung und Benutzerverwaltung für Ihre Web- und Mobilanwendungen. Ihre Benutzer können sich direkt mit einem Benutzernamen und einem Passwort oder über einen Drittanbieter wie Facebook, Amazon oder Google anmelden.

Die zwei Hauptkomponenten von Amazon Cognito sind Benutzerpools und Identitäten-Pools. Benutzerpools sind Benutzerverzeichnisse, die Registrierungs- und Anmeldungsoptionen für Ihre mobilen Anwendungs-Nutzer bereitstellen. Mithilfe von Identitätspools können Sie Ihren Benutzern

Zugriff auf andere AWS Dienste gewähren. Sie können Identitäten-Pools und Benutzerpools getrennt oder zusammen verwenden.

Weitere Informationen zu Amazon Cognito erhalten Sie im Benutzerhandbuch von Amazon Cognito.

Amazon S3: Gewährt Verbundbenutzern programmgesteuert und in der Konsole Zugriff auf ihr S3-Stammverzeichnis

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es Verbundbenutzern ermöglicht, auf ihr eigenes Startverzeichnis-Bucket-Objekt in S3 zuzugreifen. Das Stammverzeichnis ist ein Bucket mit einem home-Verzeichnis sowie Verzeichnissen für einzelne Verbundbenutzer. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Die Variable `${aws:userid}` in dieser Richtlinie wird in `role-id:specified-name` aufgelöst. Der Teil `role-id` der ID des Verbundbenutzers ist ein eindeutiger Bezeichner für die Rolle des Verbundbenutzers während der Erstellung. Weitere Informationen finden Sie unter Eindeutige Bezeichner. Der `specified-name` ist der RoleSessionName Parameter, der an die `AssumeRoleWithWebIdentity` Anfrage übergeben wurde, als der Verbundbenutzer seine Rolle übernommen hat.

Sie können die Rollen-ID mit dem AWS CLI Befehl `aws iam get-role --role-name` *`specified-name`* anzeigen. Stellen Sie sich z. B. vor, Sie geben als Anzeigenamen `John` an und die CLI gibt die Rolle-ID `AROAXXT2NJT7D3SIQN7Z6` zurück. In diesem Fall ist die ID des Verbundbenutzers `AROAXXT2NJT7D3SIQN7Z6:John`. Diese Richtlinie erlaubt dann dem Vebundbenutzer John den Zugriff auf den Amazon S3-Bucket mit dem Präfix `AROAXXT2NJT7D3SIQN7Z6:John`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
```

```
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:userid}",
                "arn:aws:s3:::bucket-name/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3: S3-Bucket-Zugriff, der Zugriff auf den Produktions-Bucket ohne vorherige MFA wird aber verweigert

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die einem Amazon-S3-Administrator den Zugriff auf jeden Bucket erlaubt, einschließlich dem Aktualisieren, Hinzufügen und Löschen von Objekten. Dabei wird jedoch explizit der Zugriff auf den `Production`-Bucket verweigert, wenn sich der Benutzer nicht innerhalb der letzten 30 Minuten per Multifaktor-Authentifizierung (MFA) angemeldet hat. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion in der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS

auszuführen. Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Diese Richtlinie ermöglicht niemals den programmgesteuerten Zugriff auf den `Production`-Bucket mit langfristig geltenden Benutzerzugriffsschlüsseln. Dies wird mit dem `aws:MultiFactorAuthAge`-Bedingungsschlüssel mit dem `NumericGreaterThanIfExists`-Bedingungsoperator erreicht. Diese Richtlinienbedingung gibt `true` zurück, wenn MFA nicht vorhanden ist oder wenn das Alter des MFA größer als 30 Minuten ist. In diesen Situationen wird der Zugriff verweigert. Um programmgesteuert auf den `Production` Bucket zuzugreifen, muss der S3-Administrator temporäre Anmeldeinformationen verwenden, die in den letzten 30 Minuten mithilfe des API-Vorgangs generiert wurden. GetSessionToken

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
```

```
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::Production/*",
                "arn:aws:s3:::Production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3:: Gewährt IAM-Benutzern programmgesteuert und in der Konsole Zugriff auf ihr S3-Stammverzeichnis.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es IAM-Benutzern erlaubt, auf ihr eigenes Startverzeichnis-Bucket-Objekt in S3 zuzugreifen. Das Stammverzeichnis ist ein Bucket mit dem Verzeichnis `home` sowie Verzeichnissen für einzelne Benutzer. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Diese Richtlinie funktioniert nicht bei der Verwendung von IAM-Rollen, da die `aws:username`-Variable bei der Verwendung von IAM-Rollen nicht verfügbar ist. Weitere Informationen zu Hauptschlüsselwerten finden Sie unter Auftraggeber-Schlüsselwerte.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
```

```
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:username}",
                "arn:aws:s3:::bucket-name/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3: Beschränken der Verwaltung auf einen bestimmten S3-Bucket

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Verwaltung eines Amazon-S3-Buckets auf diesen spezifischen Bucket beschränkt. Diese Richtlinie gewährt die Berechtigung, alle Amazon-S3-Aktionen durchzuführen, verweigert aber den Zugriff auf jeden AWS-Service außer Amazon S3. Sehen Sie sich das folgende -Beispiel an. Gemäß dieser Richtlinie können Sie nur auf Amazon-S3-Aktionen zugreifen, die Sie für einen S3-Bucket oder eine S3-Objektressource durchführen können. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese

Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Wenn diese Richtlinie in Kombination mit anderen Richtlinien (wie den FullAccess AWS verwalteten AmazonS3 FullAccess - oder AmazonEC2-Richtlinien) verwendet wird, die Aktionen zulassen, die durch diese Richtlinie verweigert wurden, wird der Zugriff verweigert. Der Grund hierfür liegt darin, dass eine explizite Zugriffsverweigerung Vorrang vor einer Zugriffserlaubnis hat. Weitere Informationen finden Sie unter the section called “Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird”.

> **Warning**
>
> Bei `NotAction` und `NotResource` handelt es sich um erweiterte Richtlinienelemente, die mit Vorsicht verwendet werden sollten. Diese Richtlinie verweigert den Zugriff auf alle AWS -Services mit Ausnahme von Amazon S3. Wenn Sie diese Richtlinie einem Benutzer anfügen, werden alle anderen Richtlinien, die Berechtigungen für andere Services erteilen, ignoriert und der Zugriff wird verweigert.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
```

```
}
```

Amazon S3: Ermöglicht Lese- und Schreibzugriff auf Objekte in einem S3-Bucket

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die `Read` und `Write` den Zugriff auf Objekte in einem bestimmten S3-Bucket erlaubt. Diese Richtlinie gewährt die erforderlichen Berechtigungen, um diese Aktion programmgesteuert über die AWS API oder abzuschließen. AWS CLI Um diese Richtlinie zu verwenden, ersetzen Sie den *kursiv gedruckten Platzhaltertext* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Die `s3:*Object`-Aktion verwendet einen Platzhalter als Teil des Namens der Aktion. Die Anweisung `AllObjectActions` erlaubt die Aktionen `GetObject`, `DeleteObject`, `PutObject` und alle anderen Amazon S3-Aktionen, die mit dem Wort "Object" enden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> Informationen zum Erlauben von `Read`- und `Write`-Zugriff auf ein Objekt in einem Amazon S3-Bucket und das Einschließen zusätzlicher Berechtigungen für den Konsolenzugriff finden Sie unter Amazon S3: Gewährt Lese- und Schreibzugriff auf Objekte in einem S3-Bucket programmgesteuert und in der Konsole.

Amazon S3: Gewährt Lese- und Schreibzugriff auf Objekte in einem S3-Bucket programmgesteuert und in der Konsole

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die `Read` und `Write` den Zugriff auf Objekte in einem bestimmten S3-Bucket erlaubt. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

Die `s3:*Object`-Aktion verwendet einen Platzhalter als Teil des Namens der Aktion. Die Anweisung `AllObjectActions` erlaubt die Aktionen `GetObject`, `DeleteObject`, `PutObject` und alle anderen Amazon S3-Aktionen, die mit dem Wort "Object" enden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
```

```
        }
    ]
}
```

Verwalten von IAM-Richtlinien

IAM gibt Ihnen die Werkzeuge an die Hand, um alle Arten von IAM-Richtlinien (verwaltete Richtlinien und Inline-Richtlinien) zu erstellen und zu verwalten. Um einer IAM-Identität (IAM-Benutzer, -Gruppe oder -Rolle) Berechtigungen hinzuzufügen, erstellen Sie eine Richtlinie, validieren die Richtlinie und fügen die Richtlinie dann der Identität zu. Sie können mehrere Richtlinien an eine Identität anfügen, wobei jede Richtlinie mehrere Berechtigungen enthalten kann.

Nutzen Sie diese Ressourcen für weitere Informationen:

- Weitere Informationen zu den unterschiedlichen IAM-Richtlinienarten finden Sie unter Berechtigungen und Richtlinien in IAM.
- Allgemeine Informationen über die Verwendung von Richtlinien innerhalb von IAM finden Sie unter Zugriffsmanagement für AWS Ressourcen.
- Weitere Informationen zum Auswerten von Berechtigungen, wenn mehrere Richtlinien für eine bestimmte IAM-Identität gelten, finden Sie unter Auswertungslogik für Richtlinien.
- Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Themen

Erstellen von IAM-Richtlinien

Eine Richtlinie ist eine Entität, die, angefügt an eine Identität oder Ressource, deren Berechtigungen definiert. Sie können die AWS API AWS Management Console, oder verwenden AWS CLI, um vom Kunden verwaltete Richtlinien in IAM zu erstellen. Vom Kunden verwaltete Richtlinien sind eigenständige Richtlinien, die Sie in Ihrem eigenen AWS-Konto verwalten. Anschließend können Sie die Richtlinien den Identitäten (Benutzern, Gruppen und Rollen) in Ihrem hinzufügen. AWS-Konto

Eine Richtlinie, die an eine Identität in IAM angefügt ist, wird als identitätsbasierte Richtlinie bezeichnet. Identitätsbasierte Richtlinien können verwaltete Richtlinien, vom Kunden AWS verwaltete Richtlinien und Inline-Richtlinien umfassen. AWS verwaltete Richtlinien werden von erstellt und verwaltet. AWS Sie können sie verwenden, aber Sie können sie nicht verwalten. Eine eingebundene Richtlinie ist eine Richtlinie, die Sie erstellen und direkt in eine Gruppe, einen Benutzer oder eine Rolle von IAM einbetten. Inline-Richtlinien können nicht für andere Identitäten wiederverwendet oder außerhalb der Identität verwaltet werden, wo sie vorhanden sind. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

Verwenden von vom Kunden verwalteten Richtlinien anstelle von eingebundenen Richtlinien. Es empfiehlt sich auch, vom Kunden verwaltete Richtlinien anstelle von AWS verwalteten Richtlinien zu verwenden. AWS Verwaltete Richtlinien bieten in der Regel umfassende administrative oder schreibgeschützte Berechtigungen. Gewähren Sie für höchste Sicherheit die geringsten Rechte, durch die nur die Berechtigungen erteilt werden, die zum Ausführen bestimmter Auftragsaufgaben erforderlich sind.

Wenn Sie IAM-Richtlinien erstellen oder bearbeiten, AWS kann es automatisch eine Richtlinienvalidierung durchführen, um Ihnen bei der Erstellung einer effektiven Richtlinie mit den geringsten Rechten zu helfen. In der AWS Management Console identifiziert IAM JSON-Syntaxfehler, während IAM Access Analyzer zusätzliche Richtlinienprüfungen mit Empfehlungen bietet, mit denen Sie Ihre Richtlinien weiter verfeinern können. Weitere Informationen zur Richtlinienvalidierung finden Sie unter Validieren von IAM-Richtlinien. Weitere Informationen zu den Richtlinienvalidierungen von IAM Access Analyzer Richtlinien und Empfehlungen erhalten Sie unter Richtlinienvalidierung von IAM Access Analyzer.

Sie können die AWS API, oder verwenden AWS Management Console AWS CLI, um vom Kunden verwaltete Richtlinien in IAM zu erstellen. Weitere Informationen zur Verwendung von AWS CloudFormation Vorlagen zum Hinzufügen oder Aktualisieren von Richtlinien finden Sie in der Referenz zum AWS Identity and Access Management Ressourcentyp im AWS CloudFormation Benutzerhandbuch.

Themen

Erstellen von IAM-Richtlinien (Konsole)

Eine Richtlinie ist eine Entität, die, angefügt an eine Identität oder Ressource, deren Berechtigungen definiert. Sie können die verwenden AWS Management Console , um vom Kunden verwaltete Richtlinien in IAM zu erstellen. Vom Kunden verwaltete Richtlinien sind eigenständige Richtlinien, die Sie in Ihrem eigenen AWS-Konto verwalten. Anschließend können Sie die Richtlinien an Identitäten (Benutzer, Gruppen und Rollen) in Ihrem anhängen. AWS-Konto

Themen

Erstellen von IAM-Richtlinien

Sie können eine vom Kunden verwaltete Richtlinie AWS Management Console mithilfe einer der folgenden Methoden erstellen:

- JSON – Fügen Sie eine veröffentlichte identitätsbasierte Beispielrichtlinie ein und passen Sie sie an.
- Visueller Editor – Sie können eine neue Richtlinie von Grund auf im visuellen Editor erstellen. Wenn Sie den visuellen Editor verwenden, müssen Sie nicht mit der JSON-Syntax vertraut sein.
- Importieren – Importieren Sie eine verwaltete Richtlinie aus Ihrem Konto und passen Sie sie an. Sie können eine AWS verwaltete Richtlinie oder eine vom Kunden verwaltete Richtlinie importieren, die Sie zuvor erstellt haben.

Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Erstellen von Richtlinien mit dem JSON-Editor

Sie können Richtlinien in JSON eingeben oder einfügen, indem Sie die Option JSON auswählen. Diese Methode ist nützlich für das Kopieren einer Beispielrichtlinie zur Verwendung in Ihrem Konto. Alternativ können Sie Ihr eigenes JSON-Richtliniendokument im JSON-Editor eingeben. Sie können auch die Option JSON verwenden, um zwischen dem Visual-Editor und JSON zu wechseln und so die Ansichten zu vergleichen.

Wenn Sie eine Richtlinie im JSON-Editor erstellen oder bearbeiten, führt IAM eine Richtlinienvalidierung durch, um Ihnen beim Erstellen einer effektiven Richtlinie zu helfen. IAM identifiziert JSON-Syntaxfehler, während IAM Access Analyzer zusätzliche Richtlinienüberprüfungen mit umsetzbaren Empfehlungen zur weiteren Verfeinerung der Richtlinie bereitstellt.

Das Dokument einer JSON-Richtlinie besteht aus mindestens einer Anweisung. Jede Anweisung sollte alle Aktionen mit den gleichen Auswirkungen (`Allow` oder `Deny`) enthalten und dieselben Ressourcen und Bedingungen unterstützen. Wenn eine Aktion es erforderlich macht, dass Sie alle Ressourcen angeben (`"*"`) und eine andere Aktion den Amazon-Ressourcennamen (ARN) einer bestimmten Ressource unterstützt, sind für diese Aktionen zwei separate JSON-Anweisungen erforderlich. Weitere Informationen zu ARN-Formaten finden Sie unter Amazon-Ressourcenname (ARN) im Allgemeine AWS-Referenz Handbuch. Weitere allgemeine Informationen zu IAM-Richtlinien finden Sie unter Berechtigungen und Richtlinien in IAM. Informationen zur IAM-Richtliniensprache finden Sie unter IAM-JSON-Richtlinienreferenz.

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).
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben oder fügen Sie ein JSON-Richtliniendokument ein. Weitere Informationen zur IAM-Richtliniensprache finden Sie unter IAM-JSON-Richtlinienreferenz.
6. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

7. (Optional) Wenn Sie eine Richtlinie in der erstellen oder bearbeiten AWS Management Console, können Sie eine JSON- oder YAML-Richtlinienvorlage generieren, die Sie in AWS CloudFormation Vorlagen verwenden können.

   Wählen Sie dazu im Richtlinien-Editor Aktionen und anschließend CloudFormationVorlage generieren aus. Weitere Informationen dazu AWS CloudFormation finden Sie in der Referenz zum AWS Identity and Access Management Ressourcentyp im AWS CloudFormation Benutzerhandbuch.
8. Wenn Sie mit dem Hinzufügen von Berechtigungen zur Richtlinie fertig sind, wählen Sie Next (Weiter) aus.
9. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) unter Name einen Namen und unter Description (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.
10. (Optional) Fügen Sie der Richtlinie Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.
11. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Nachdem Sie eine Richtlinie erstellt haben, können Sie sie an Ihre Benutzer, Gruppen oder Rollen anfügen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

Erstellen von Richtlinien mit dem visuellen Editor

Der visuelle Editor in der IAM-Konsole führt Sie durch das Erstellen einer Richtlinie, wobei das Schreiben von JSON-Syntax nicht erforderlich ist. Ein Beispiel für die Verwendung des visuellen Editors zum Erstellen einer Richtlinie finden Sie unter the section called “Steuern des Zugriffs auf Identitäten”.

So verwenden Sie den visuellen Editor 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).
3. Wählen Sie Richtlinie erstellen aus.
4. Suchen Sie im Bereich Richtlinien-Editor nach dem Abschnitt Dienst auswählen und wählen Sie dann einen AWS Dienst aus. Sie können das Suchfeld oben verwenden, um die Ergebnisse in der Liste der Services einzuschränken. Sie können nur einen Service innerhalb eines Berechtigungsblocks des visuellen Editors auswählen. Um mehr als einem Service Zugriff zu gewähren, fügen Sie mehrere Berechtigungsblöcke hinzu, indem Sie Add more permissions (Weitere Berechtigungen hinzufügen) auswählen.
5. Wählen Sie unter Actions allowed (Zulässige Aktionen) die Aktionen aus, die der Richtlinie hinzugefügt werden sollen. Es gibt folgende Möglichkeiten, Aktionen auszuwählen:

   - Markieren Sie das Kontrollkästchen für alle Aktionen.
   - Wählen Sie Add actions (Aktionen hinzufügen), um den Namen einer bestimmten Aktion einzugeben. Sie können Platzhalter (*) verwenden, um mehrere Aktionen anzugeben.
   - Wählen Sie eine der Access level ((Zugriffsebene)-Gruppen aus, um alle Aktionen für die Zugriffsebene auszuwählen (z. B. Read (Lesen), Write (Schreiben) oder List (Auflisten).
   - Erweitern Sie die einzelnen Gruppen Access level (Zugriffsebene), um einzelne Aktionen auszuwählen.

   Standardmäßig lässt die Richtlinie, die Sie erstellen, die Aktionen zu, die Sie auswählen. Um die ausgewählten Aktionen stattdessen zu verweigern, wählen Sie Switch to deny permissions (Zu Berechtigungen verweigern wechseln). Da IAM standardmäßig verweigert, empfehlen wir, dass Sie im Sinne bewährter Sicherheitsmethoden nur für jene Aktionen und Ressourcen Berechtigungen zulassen, für die ein Benutzer Zugriff benötigt. Sie sollten nur dann eine JSON-Anweisung erstellen, um Berechtigungen zu verweigern, wenn Sie eine von einer anderen Anweisung oder Richtlinie erteilte Berechtigung separat überschreiben möchten. Wir empfehlen, die Anzahl der Verweigerungsberechtigungen auf ein Minimum zu beschränken, da diese die Fehlerbehebung bei Berechtigungen erschweren.
6. Wenn bei Resources (Ressourcen) der Service und die Aktionen, die Sie in den vorherigen Schritten ausgewählt haben, nicht die Auswahl bestimmter Ressourcen unterstützen, sind alle Ressourcen zulässig, und Sie können diesen Abschnitt nicht bearbeiten.

Wenn Sie eine oder mehrere Aktionen auswählen, die Berechtigungen auf Ressourcenebene unterstützen, dann listet der visuelle Editor diese Ressourcen auf. Sie können dann Resources (Ressourcen) erweitern, um die Ressourcen für Ihre Richtlinie anzugeben.

Sie können Ressourcen auf folgende Weise angeben:

- Wählen Sie Add ARNs (ARNs hinzufügen) aus, um Ressourcen anhand ihres Amazon-Ressourcennamens (ARN) anzugeben. Sie können den visuellen ARN-Editor verwenden oder ARNs manuell auflisten. Weitere Informationen zur ARN-Syntax finden Sie unter Amazon-Ressourcenname (ARN) im Allgemeine AWS-Referenz -Handbuch. Hinweise zur Verwendung von ARNs im `Resource`-Element einer Richtlinie finden Sie unter IAM-JSON-Richtlinienelemente: Resource.
- Wählen Sie Any in this account (Alle in diesem Konto) neben einer Ressource aus, um Berechtigungen für alle Ressourcen dieses Typs zu gewähren.
- Wählen Sie All (Alle) aus, um alle Ressourcen für den Service auszuwählen.

7. (Optional) Wählen Sie Request conditions - optional (Anfragebedingungen - (optional)) aus, um der Richtlinie, die Sie erstellen, Bedingungen hinzuzufügen. Bedingungen schränken die Auswirkungen einer JSON-Richtlinienanweisung ein. Sie können beispielsweise festlegen, dass einem Benutzer erlaubt wird, die Aktionen für die Ressourcen nur durchzuführen, wenn die Anforderung dieses Benutzers in einem bestimmten Zeitraum stattfindet. Sie können auch allgemein genutzte Bedingungen verwenden, um festzulegen, ob ein Benutzer mithilfe eines Multi-Factor Authentication (MFA)-Geräts authentifiziert werden muss. Oder Sie können festlegen, dass die Anforderung aus einem bestimmten IP-Adressbereich stammen muss. Eine Liste aller Kontextschlüssel, die Sie in einer Richtlinienbedingung verwenden können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste in der Service Authorization Reference.

   Sie haben folgende Möglichkeiten, Bedingungen auszuwählen:

   - Verwenden Sie Kontrollkästchen, um allgemein verwendete Bedingungen auszuwählen.
   - Wählen Sie Add another condition (Weitere Bedingung hinzufügen) aus, um andere Bedingungen anzugeben. Wählen Sie den Condition Key (Bedingungsschlüssel), den Qualifier (Qualifizierer) und den Operator der Bedingung aus und geben Sie dann einen Value (Wert) ein. Um mehr als einen Wert hinzuzufügen, wählen Sie Add (Hinzufügen) aus. Betrachten Sie die Werte, als wären sie mit einem logischen "ODER"-Operator miteinander verbunden. Wählen Sie danach Add condition (Bedingung hinzufügen) aus.

Um mehr als eine Bedingung hinzuzufügen, wählen Sie Add another condition (Weitere Bedingung hinzufügen) aus. Wiederholen Sie diesen Vorgang nach Bedarf. Jede Bedingung gilt nur für diesen einen Berechtigungsblock des visuellen Editors. Alle Bedingungen müssen wahr sein, damit der Berechtigungsblock ausgeführt werden kann. Anders ausgedrückt: Betrachten Sie die Bedingungen als durch einen logischen "UND"-Operator miteinander verbunden.

Weitere Informationen zum Element Condition (Bedingung) finden Sie unter IAM-JSON-Richtlinienelemente: Condition im IAM-JSON-Richtlinienreferenz.

8. Um mehr Berechtigungsblöcke hinzuzufügen, wählen Sie Add more permissions (Weitere Berechtigungen hinzufügen) aus. Wiederholen Sie die Schritte 2 bis 5 für jeden Block.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

9. (Optional) Wenn Sie eine Richtlinie in der erstellen oder bearbeiten AWS Management Console, können Sie eine JSON- oder YAML-Richtlinienvorlage generieren, die Sie in AWS CloudFormation Vorlagen verwenden können.

   Wählen Sie dazu im Richtlinien-Editor Aktionen und anschließend CloudFormationVorlage generieren aus. Weitere Informationen dazu AWS CloudFormation finden Sie in der Referenz zum AWS Identity and Access Management Ressourcentyp im AWS CloudFormation Benutzerhandbuch.

10. Wenn Sie mit dem Hinzufügen von Berechtigungen zur Richtlinie fertig sind, wählen Sie Next (Weiter) aus.

11. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) unter Name einen Namen und unter Description (Beschreibung) (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um sicherzustellen, dass Sie die beabsichtigten Berechtigungen erteilt haben.

12. (Optional) Fügen Sie der Richtlinie Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen.

13. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Nachdem Sie eine Richtlinie erstellt haben, können Sie sie an Ihre Benutzer, Gruppen oder Rollen anfügen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

Importieren vorhandener verwalteter Richtlinien

Eine einfache Möglichkeit zum Erstellen einer neuen Richtlinie besteht darin, eine vorhandene verwaltete Richtlinie innerhalb Ihres Kontos zu importieren, die mindestens einige der Berechtigungen aufweist, die Sie benötigen. Sie können dann die Richtlinie an Ihre neuen Anforderungen anpassen.

Sie können keine eingebundene Richtlinie importieren. Informationen über den Unterschied zwischen verwalteten und eingebundenen Richtlinien finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien.

So importieren Sie eine vorhandene verwaltete Richtlinie im visuellen Editor

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).
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie im Policy editor (Richtlinien-Editor) Visual aus und dann rechts auf der Seite die Option Actions (Aktionen) und anschließend Import policy (Richtline importieren) aus.
5. Wählen Sie im Fenster Import policy (Richtlinie importieren) die verwalteten Richtlinien aus, die am ehesten mit der Richtlinie übereinstimmen, die Sie in Ihre neue Richtlinie einschließen möchten. Sie können das Suchfeld oben verwenden, um die Ergebnisse in der Liste der Richtlinien einzuschränken.
6. Wählen Sie Import policy (Richtlinie importieren) aus.

   Die importierten Richtlinien werden in neuen Berechtigungsblöcken unten in der Richtlinie hinzugefügt.
7. Verwenden Sie den Visual editor (Visueller Editor) oder wählen Sie JSON aus, um Ihre Richtlinie anzupassen. Wählen Sie anschließend Weiter.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

8. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) unter Name einen Namen und unter Description (Beschreibung) (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Sie können diese Einstellungen später nicht bearbeiten. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen) und wählen Sie dann Create policy (Richtline erstellen) aus, um Ihre Eingaben zu speichern.

So importieren Sie eine vorhandene verwaltete Richtlinie in den JSON-Editor

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).
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie im Abschnitt Policy editor (Richtlinien-Editor) die Option JSON aus und dann rechts auf der Seite die Option Actions (Aktionen) und anschließend Import policy (Richtline importieren) aus.
5. Wählen Sie im Fenster Import policy (Richtlinie importieren) die verwalteten Richtlinien aus, die am ehesten mit der Richtlinie übereinstimmen, die Sie in Ihre neue Richtlinie einschließen möchten. Sie können das Suchfeld oben verwenden, um die Ergebnisse in der Liste der Richtlinien einzuschränken.
6. Wählen Sie Import policy (Richtlinie importieren) aus.

   Anweisungen aus den importierten Richtlinien werden unten in Ihrer JSON-Richtlinie hinzugefügt.
7. Passen Sie Ihre Richtlinie in JSON an. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter. Passen Sie Ihre Richtlinie in JSON an oder wählen Sie den Visual editor (Visueller Editor). Wählen Sie anschließend Weiter.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

8. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) unter Name einen Namen und unter Description (Beschreibung) (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Sie können diese später nicht mehr bearbeiten. Überprüfen Sie die Richtlinie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen) und wählen Sie dann Create policy (Richtline erstellen) aus, um Ihre Eingaben zu speichern.

Nachdem Sie eine Richtlinie erstellt haben, können Sie sie an Ihre Benutzer, Gruppen oder Rollen anfügen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.

Erstellen von IAM-Richtlinien (AWS CLI)

Eine Richtlinie ist eine Entität, die, angefügt an eine Identität oder Ressource, deren Berechtigungen definiert. Sie können die verwenden AWS CLI , um vom Kunden verwaltete Richtlinien in IAM zu erstellen. Vom Kunden verwaltete Richtlinien sind eigenständige Richtlinien, die Sie in Ihrem eigenen AWS-Konto verwalten. Als bewährte Methode empfehlen wir, IAM Access Analyzer zu verwenden, um Ihre IAM-Richtlinien zu validieren und sichere und funktionale Berechtigungen zu gewährleisten. Indem Sie Ihre Richtlinien validieren, können Sie Fehler oder Empfehlungen beheben, bevor Sie die Richtlinien an Identitäten (Benutzer, Gruppen und Rollen) in Ihrem AWS-Konto anfügen.

Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Erstellen von IAM-Richtlinien (AWS CLI)

Sie können eine vom Kunden verwaltete IAM-Richtlinie oder eine Inline-Richtlinie mithilfe der AWS Command Line Interface (AWS CLI) erstellen.

So erstellen Sie eine kundenverwaltete Richtlinie (AWS CLI)

Verwenden Sie den folgenden Befehl:

- create-policy

So erstellen Sie eine Inline-Richtlinie für eine IAM-Identität (Gruppe, Benutzer oder Rolle) (AWS CLI)

Verwenden Sie einen der folgenden Befehle:

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> Sie können IAM nicht verwenden, um eine Inline-Richtlinie für eine dienstverknüpfte Rolle einzubetten.

So bearbeiten Sie eine kundenverwaltete Richtlinie (AWS CLI)

Verwenden Sie den folgenden IAM Access Analyzer-Befehl:

- validieren-policy

IAM-Richtlinien (AWS API) erstellen

Eine Richtlinie ist eine Entität, die, angefügt an eine Identität oder Ressource, deren Berechtigungen definiert. Sie können die AWS API verwenden, um vom Kunden verwaltete Richtlinien in IAM zu erstellen. Vom Kunden verwaltete Richtlinien sind eigenständige Richtlinien, die Sie in Ihrem eigenen AWS-Konto verwalten. Als bewährte Methode empfehlen wir, IAM Access Analyzer zu verwenden, um Ihre IAM-Richtlinien zu validieren und sichere und funktionale Berechtigungen zu gewährleisten. Indem Sie Ihre Richtlinien validieren, können Sie Fehler oder Empfehlungen beheben, bevor Sie die Richtlinien an Identitäten (Benutzer, Gruppen und Rollen) in Ihrem AWS-Konto anfügen.

Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

IAM-Richtlinien (API)AWS erstellen

Mit der AWS -API können Sie eine vom IAM-Kunden verwaltete -Richtlinie oder eine Inline-Richtlinie erstellen.

Um eine vom Kunden verwaltete Richtlinie (AWS API) zu erstellen

Rufen Sie die folgende Operation auf:

- CreatePolicy

So erstellen Sie eine Inline-Richtlinie für eine IAM-Identität (Gruppe, Benutzer oder Rolle) (AWS -API)

Rufen Sie eine der folgenden Operationen auf:

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> Sie können IAM nicht verwenden, um eine Inline-Richtlinie für eine dienstverknüpfte Rolle einzubetten.

Um eine vom Kunden verwaltete Richtlinie (AWS API) zu validieren

Rufen Sie den folgenden IAM Access Analyzer-Vorgang auf:

- ValidatePolicy

Validieren von IAM-Richtlinien

Eine Richtlinie ist ein JSON-Dokument, das mithilfe der IAM-Richtlinienformulierung erstellt wird. Wenn Sie eine Richtlinie an eine IAM-Entität, z. B. einen Benutzer, eine Gruppe oder eine Rolle, anfügen, gewährt diese Entität Berechtigungen.

Wenn Sie IAM-Zugriffskontrollrichtlinien mithilfe von erstellen oder bearbeiten AWS Management Console, AWS werden diese automatisch überprüft, um sicherzustellen, dass sie der IAM-Richtliniengrammatik entsprechen. Wenn AWS die Richtlinienvalidierung feststellt, dass eine Richtlinie nicht mit der Formulierung übereinstimmt, werden Sie aufgefordert, die Richtlinie zu korrigieren.

IAM Access Analyzer bietet zusätzliche Richtlinienüberprüfungen mit Empfehlungen, die Ihnen helfen, die Richtlinie weiter zu verfeinern. Weitere Informationen zu den Richtlinienvalidierungen von IAM Access Analyzer Richtlinien und Empfehlungen erhalten Sie unter Richtlinienvalidierung von IAM Access Analyzer. Eine Liste der Warnungen, Fehler und Vorschläge, die von IAM Access Analyzer zurückgegeben werden, finden Sie unter IAM-Access-Analyzer-Richtlinienprüfungsreferenz.

Gültigkeitsumfang

AWS überprüft die Syntax und Grammatik der JSON-Richtlinie. Außerdem wird überprüft, ob Ihre ARNs korrekt formatiert sind und Aktionsnamen und Bedingungsschlüssel korrekt sind.

Zugriffsrichtlinie

Richtlinien werden automatisch validiert, wenn Sie eine JSON-Richtlinie erstellen oder eine vorhandene Richtlinie in der AWS Management Console bearbeiten. Wenn die Richtlinie nicht gültig ist, erhalten Sie eine Benachrichtigung und müssen das Problem beheben, bevor Sie fortfahren können. Die Ergebnisse der IAM Access Analyzer-Richtlinienvalidierung werden automatisch zurückgegeben, AWS Management Console sofern Sie über Berechtigungen für `access-analyzer:ValidatePolicy` verfügen. Sie können Richtlinien auch mithilfe der AWS API oder AWS CLIüberprüfen.

Bestehende Richtlinien

Möglicherweise verfügen Sie über vorhandene Richtlinien, die nicht gültig sind, da sie erstellt oder zuletzt gespeichert wurden, bevor die letzten Updates des Richtlinienmoduls aktualisiert wurden. Als bewährte Methode empfehlen wir, IAM Access Analyzer zu verwenden, um Ihre IAM-Richtlinien zu validieren und sichere und funktionale Berechtigungen zu gewährleisten. Wir empfehlen Ihnen, Ihre vorhandenen Richtlinien zu öffnen und die generierten Richtlinien-Validierungsergebnisse zu überprüfen. Sie können vorhandene Richtlinien nicht bearbeiten und speichern, ohne die Syntaxfehler der Richtlinien zu beheben.

Generieren von Richtlinien basierend auf Zugriffsaktivitäten

Als Administrator oder Entwickler können Sie IAM-Entitäten (Benutzern oder Rollen) Berechtigungen gewähren, die über das hinausgehen, was sie benötigen. IAM bietet verschiedene Optionen, mit denen Sie die von Ihnen erteilten Berechtigungen verfeinern können. Eine Option besteht darin, eine IAM-Richtlinie zu generieren, die auf der Zugriffsaktivität für eine Entität basiert. IAM Access Analyzer überprüft Ihre AWS CloudTrail Protokolle und generiert eine Richtlinienvorlage, die die Berechtigungen enthält, die die Entität in Ihrem angegebenen Zeitraum verwendet hat. Sie können

die Vorlage verwenden, um eine Richtlinie mit definierten Berechtigungen zu erstellen, die nur die Berechtigungen gewährt, die zur Unterstützung Ihres spezifischen Anwendungsfalls erforderlich sind.

Stellen Sie sich zum Beispiel vor, Sie sind ein Entwickler und Ihr Ingenieurteam arbeitet an einem Projekt zur Erstellung einer neuen Anwendung. Um das Experimentieren zu fördern und Ihrem Team zu ermöglichen, schnell voranzukommen, haben Sie während der Entwicklung der Anwendung eine Rolle mit umfassenden Berechtigungen konfiguriert. Jetzt ist die Anwendung bereit für die Produktion. Bevor die Anwendung im Produktionskonto gestartet werden kann, möchten Sie nur die Berechtigungen identifizieren, die die Rolle benötigt, damit die Anwendung funktioniert. Auf diese Weise können Sie die bewährten Methoden der Gewährung der geringsten Rechte befolgen. Sie können eine Richtlinie basierend auf der Zugriffsaktivität der Rolle generieren, die Sie für die Anwendung im Entwicklungskonto verwendet haben. Sie können die generierte Richtlinie weiter verfeinern und die Richtlinie dann an eine Entität in Ihrem Produktionskonto anfügen.

Weitere Informationen zur Erstellung von IAM Access Analyzer finden Sie unter Erstellen von IAM Access Analyzer-Richtlinien.

Testen von IAM-Richtlinien mit dem IAM-Richtliniensimulator

Weitere Informationen dazu, wie und warum IAM-Richtlinien verwendet werden, finden Sie unter Berechtigungen und Richtlinien in IAM.

Sie erhalten Zugriff auf die IAM-Richtliniensimulator-Konsole unter: https://policysim.aws.amazon.com/

> **Important**
>
> Die Ergebnisse des Richtliniensimulators können von Ihrer AWS Live-Umgebung abweichen. Wir empfehlen Ihnen, Ihre Richtlinien nach dem Testen mit dem Richtliniensimulator mit Ihrer AWS Live-Umgebung zu vergleichen, um sicherzustellen, dass Sie die gewünschten Ergebnisse erzielt haben. Weitere Informationen finden Sie unter Funktionsweise des IAM-Richtliniensimulators.

Erste Schritte mit dem IAM-Richtliniensimulator

Mit dem IAM-Richtliniensimulator können Sie identitätsbasierte Richtlinien sowie IAM-Berechtigungsgrenzen testen und diesbezüglichen Probleme beheben. Im Folgenden finden Sie einige gängige Aktionen, die Sie mit dem Richtliniensimulator ausführen können:

- Testen Sie die identitätsbasierten Richtlinien, die mit IAM-Benutzern, -Gruppen oder IAM-Rollen in Ihrem AWS-Konto-Konto verknüpft sind. Wenn mehrere Richtlinie mit dem Benutzer, der Gruppe oder der Rolle verknüpft sind, können Sie alle Richtlinien testen oder einzelne Richtlinien für den Test auswählen. Sie können testen, welche Aktionen die ausgewählten Richtlinien für bestimmte Ressourcen zulassen oder verweigern.
- Testen und beheben Sie die Auswirkungen von Berechtigungsgrenzen auf IAM-Entitäten. Hinweis: Sie können jeweils nur eine Berechtigungsgrenze simulieren.
- Testen Sie die Auswirkungen der ressourcenbasierten Richtlinien zu IAM-Benutzern, die mit AWS -Ressourcen verbunden sind, wie Amazon-S3-Buckets, Amazon-SQS-Warteschlangen, Amazon-SNS-Themen oder Amazon-S3 Glacier-Datenspeicher. Um eine ressourcenbasierte Richtlinie im Richtliniensimulator für IAM-Benutzer zu verwenden, müssen Sie die Ressource in die Simulation aufnehmen. Sie müssen außerdem das Kontrollkästchen aktivieren, um die Richtlinie dieser Ressource in die Simulation aufzunehmen.

  > **Note**
  >
  > Die Simulation ressourcenbasierter Richtlinien wird für IAM-Rollen nicht unterstützt.

- Wenn Sie AWS-Konto Mitglied einer Organisation in sind, können Sie testen AWS Organizations, wie sich Service Control Policies (SCPs) auf Ihre identitätsbasierten Richtlinien auswirken.

  > **Note**
  >
  > Der Richtliniensimulator bewertet keine SCPs mit allen Bedingungen.

- Testen Sie neue identitätsbasierten Richtlinien, die noch nicht mit einem Benutzer, einer Gruppe oder einer Rolle verknüpft sind, indem Sie sie in den Richtliniensimulator eingeben oder dorthin kopieren. Diese werden nur in der Simulation verwendet, aber nicht gespeichert. Sie können eine ressourcenbasierte Richtlinie nicht in den Richtliniensimulator eingeben oder dorthin kopieren.
- Testen Sie identitätsbasierte Richtlinien mit ausgewählten Services, Aktionen und Ressourcen. Sie können beispielsweise Ihre Richtlinie dahingehend testen, um sicherzustellen, dass sie einer Entität ermöglicht, die Aktionen `ListAllMyBuckets`, `CreateBucket` und `DeleteBucket` im Amazon S3-Service für einen bestimmten Bucket auszuführen.
- Simulieren Sie realitätsnahe Szenarien durch Bereitstellen von Kontextschlüsseln, z. B. eine IP-Adresse oder ein Datum, die in den `Condition`-Elementen der zu testenden Richtlinien enthalten sind.

> **Note**
>
> Der Richtliniensimulator simuliert keine als Eingabe bereitgestellten Tags, wenn die identitätsbasierte Richtlinie in der Simulation kein `Condition`-Element enthält, das explizit nach Tags sucht.

- Bestimmen Sie, welche spezifische Anweisung in einer identitätsbasierten Richtlinie zur Folge hat, dass Zugriff auf eine bestimmte Ressource oder Aktion gewährt oder verweigert wird.

Themen

Funktionsweise des IAM-Richtliniensimulators

Der Richtliniensimulator bewertet die Aussagen der identitätsbasierten Richtlinie und die Eingaben, die Sie während der Simulation angeben. Die Ergebnisse des Richtliniensimulators können von Ihrer AWS -Live-Umgebung abweichen. Wir empfehlen Ihnen, Ihre Richtlinien nach dem Testen mit dem Richtliniensimulator mit Ihrer AWS Live-Umgebung zu vergleichen, um sicherzustellen, dass Sie die gewünschten Ergebnisse erzielt haben.

Der Richtliniensimulator unterscheidet sich in folgenden Punkten von der AWS Live-Umgebung:

- Der Richtliniensimulator stellt keine tatsächliche AWS Serviceanfrage, sodass Sie Anfragen, die zu unerwünschten Änderungen an Ihrer AWS Live-Umgebung führen könnten, sicher testen können. Der Richtliniensimulator berücksichtigt die Schlüsselwerte des realen Kontextes in der Produktion nicht.
- Da der Richtliniensimulator nicht die Ausführung der ausgewählten Aktionen simuliert, kann er keine Antwort auf die simulierte Anforderung melden. Als einziges Ergebnis wird zurückgegeben, ob die angeforderte Aktion gewährt oder verweigert wird.
- Wenn Sie eine Richtlinie im Richtliniensimulator bearbeiten, wirken sich diese Änderungen nur auf den Richtliniensimulator aus. Die entsprechende Richtlinie in Ihrem AWS-Konto bleibt unverändert.
- Sie können Service-Kontrollrichtlinien (SCPs) nicht mit beliebigen Bedingungen testen.

- Der Richtliniensimulator unterstützt keine Simulation für IAM-Rollen und Benutzer für kontoübergreifenden Zugriff.

> **Note**
>
> Der IAM-Richtliniensimulator ermittelt nicht, welche Services globale Bedingungsschlüssel für die Autorisierung unterstützen. Der Richtliniensimulator erkennt beispielsweise nicht, dass ein Service `aws:TagKeys` nicht unterstützt.

Erforderliche Berechtigungen für die Verwendung des IAM-Richtliniensimulators

Sie können die Konsole oder die API des Richtliniensimulators zum Testen von Richtlinien verwenden. Standardmäßig können Konsolenbenutzer Richtlinien testen, die noch nicht mit einem Benutzer, einer Gruppe oder einer Rolle verknüpft sind, indem sie diese in den Richtliniensimulator eingeben oder dorthin kopieren. Diese Richtlinien werden nur in der Simulation verwendet und legen keine vertrauliche Informationen offen. API-Benutzer müssen die Berechtigungen zum Testen von nicht verknüpften Richtlinien besitzen. Sie können Konsolen- oder API-Benutzern erlauben, Richtlinien zu testen, die IAM-Benutzern, Benutzergruppen oder Rollen in Ihrem AWS-Konto angefügt sind. Hierzu müssen Sie die Berechtigung zum Abrufen dieser Richtlinien bereitstellen. Um ressourcenbasierte Richtlinien zu testen, müssen die Benutzer die Berechtigung zum Abrufen der Ressourcenrichtlinie besitzen.

Beispiele für Konsolen- und API-Richtlinien, mit denen ein Benutzer Richtlinien simulieren kann, finden Sie unter the section called “Beispielrichtlinien: AWS Identity and Access Management (IAM)”.

Erforderliche Berechtigungen für die Verwendung der Richtliniensimulator-Konsole

Sie können Benutzern die Berechtigung zum Testen von Richtlinien erteilen, die Ihren IAM-Benutzern, -Gruppen oder -Rollen in Ihrem AWS-Konto angefügt sind. Hierzu müssen Sie Ihren Benutzern Berechtigungen zum Abrufen dieser Richtlinien geben. Um ressourcenbasierte Richtlinien zu testen, müssen die Benutzer die Berechtigung zum Abrufen der Ressourcenrichtlinie besitzen.

Unter IAM: Zugriff auf die Richtliniensimulator-Konsole finden Sie eine Beispielrichtlinie, die die Verwendung der Richtliniensimulator-Konsole für die Richtlinien ermöglicht, die mit einem Benutzer, einer Gruppe oder einer Rolle verknüpft sind.

Eine Beispielrichtlinie, die die Verwendung der Richtliniensimulator-Konsole nur für die Benutzer mit einem bestimmten Pfad ermöglicht, finden Sie unter IAM: Zugriff auf die Richtliniensimulator-Konsole basierend auf dem Benutzerpfad.

Um eine Richtlinie zu erstellen, die die Verwendung der Richtliniensimulator-Konsole nur für einen Entitätstyp ermöglicht, führen Sie die folgenden Verfahren.

So ermöglichen Sie Konsolenbenutzern die Simulation von Richtlinien für Benutzer

Binden Sie die folgenden Aktionen in Ihrer Richtlinie ein:

- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetUser`
- `iam:GetUserPolicy`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupsForUser`
- `iam:ListGroupPolicies`
- `iam:ListUserPolicies`
- `iam:ListUsers`

So ermöglichen Sie Konsolenbenutzern die Simulation von Richtlinien für Gruppen

Binden Sie die folgenden Aktionen in Ihrer Richtlinie ein:

- `iam:GetGroup`
- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:ListAttachedGroupPolicies`
- `iam:ListGroupPolicies`
- `iam:ListGroups`

So ermöglichen Sie Konsolenbenutzern die Simulation von Richtlinien für Rollen

Binden Sie die folgenden Aktionen in Ihrer Richtlinie ein:

- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:GetRolePolicy`
- `iam:ListAttachedRolePolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`

Um ressourcenbasierte Richtlinien zu testen, müssen die Benutzer die Berechtigung zum Abrufen der Ressourcenrichtlinie besitzen.

So ermöglichen Sie Konsolenbenutzern das Testen von ressourcenbasierten Richtlinien in einem Amazon S3-Bucket

Binden Sie die folgenden Aktionen in Ihrer Richtlinie ein:

- `s3:GetBucketPolicy`

Die folgende Richtlinie verwendet diese Aktion beispielsweise, damit Konsolenbenutzer eine ressourcenbasierte Richtlinie in einem bestimmten Amazon S3-Bucket simulieren können.

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "s3:GetBucketPolicy",
            "Resource":"arn:aws:s3:::bucket-name/*"
          }
        ]
      }
```

Erforderliche Berechtigungen für die Verwendung der Richtliniensimulator-API

Die API des Richtliniensimulators GetContextKeyForCustomPolicyfunktioniert und SimulateCustomPolicyermöglicht es Ihnen, Richtlinien zu testen, die noch keinem Benutzer, keiner

Benutzergruppe oder Rolle zugeordnet sind. Um solche Richtlinien zu testen, übergeben Sie die Richtlinien als Zeichenfolgen an die API. Diese Richtlinien werden nur in der Simulation verwendet und legen keine vertrauliche Informationen offen. Sie können die API auch verwenden, um Richtlinien zu testen, die IAM-Benutzern, Benutzergruppen oder Rollen in Ihrem AWS-Konto angefügt sind. Dazu müssen Sie Benutzern Berechtigungen zum Aufrufen von GetContextKeyForPrincipalPolicyund gewähren SimulatePrincipalPolicy.

Ein Beispiel für eine Richtlinie, die derzeit die Verwendung der Richtliniensimulator-API für angefügte und nicht verknüpfte Richtlinien ermöglicht AWS-Konto, finden Sie unterIAM: Zugriff auf die Richtliniensimulator-API.

Um eine Richtlinie zu erstellen, die die Verwendung der Richtliniensimulator-API nur für einen Richtlinientyp ermöglicht, führen Sie die folgenden Verfahren.

So ermöglichen Sie API-Benutzern die Simulation von Richtlinien, die direkt als Zeichenfolgen an die API weitergeleitet werden

Binden Sie die folgenden Aktionen in Ihrer Richtlinie ein:

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

So ermöglichen Sie API-Benutzern die Simulation von Richtlinien, die mit IAM-Benutzern, -Gruppen, -Rollen oder IAM-Ressourcen verknüpft sind

Binden Sie die folgenden Aktionen in Ihrer Richtlinie ein:

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

Wenn Sie beispielsweise dem Benutzer Bob die Berechtigung zum Simulieren einer Richtlinie erteilen, die der Benutzerin Alice zugewiesen sind, gewähren Sie Bob Zugriff auf die folgende Ressource: `arn:aws:iam::777788889999:user/alice`.

Eine Beispielrichtlinie, die die Verwendung der Richtliniensimulator-API nur für die Benutzer mit einem bestimmten Pfad ermöglicht, finden Sie unter IAM: Zugriff auf die Richtliniensimulator-API basierend auf dem Benutzerpfad.

Verwenden des IAM-Richtliniensimulators (Konsole)

Standardmäßig können Benutzer Richtlinien testen, die noch nicht mit einem Benutzer, einer Gruppe oder einer Rolle verknüpft sind, indem sie diese in die Konsole des Richtliniensimulators eingeben oder dorthin kopieren. Diese Richtlinien werden nur in der Simulation verwendet und legen keine vertrauliche Informationen offen.

So testen Sie eine Richtlinie, die keinem Benutzer bzw. keiner Gruppe oder Rolle (Konsole) angefügt ist

1. Öffnen Sie die IAM-Richtliniensimulatorkonsole unter https://policysim.aws.amazon.com/.
2. Wählen Sie im Menü Mode: (Modus:) oben auf der Seite die Option New Policy (Neue Richtlinie).
3. Wählen Sie unter Policy Sandbox (Richtlinien-Sandbox) die Option Create New Policy (Neue Richtlinie erstellen).
4. Geben Sie eine Richtlinie in den Richtliniensimulator ein bzw. kopieren Sie sie dorthin und verwenden Sie den Richtliniensimulator wie in den folgenden Schritten beschrieben.

Sobald Sie über die Berechtigung zur Verwendung der IAM-Richtliniensimulator-Konsole verfügen, können Sie mithilfe des Richtliniensimulators einen IAM-Benutzer, eine Gruppe, Rolle oder Ressourcenrichtlinie testen.

So testen Sie eine Richtlinie, die einem Benutzer bzw. einer Gruppe oder Rolle (Konsole) angefügt ist

1. Öffnen Sie die IAM-Richtliniensimulatorkonsole unter https://policysim.aws.amazon.com/.

   > **Note**
   >
   > Zum Anmelden beim Richtliniensimulator als IAM-Benutzer verwenden Sie Ihre eindeutige Anmelde-URL, um sich bei der AWS Management Console anzumelden. Rufen Sie dann die Website https://policysim.aws.amazon.com/ auf. Weitere Informationen zum Anmelden als IAM-Benutzer finden Sie unter So melden sich IAM-Benutzer an AWS.

   Der Richtliniensimulator wird im Modus Existing Policies (Existierende Richtlinien) geöffnet und zeigt unter Users, Groups, and Roles (Benutzer, Gruppen und Rollen) die IAM-Benutzer in Ihrem Konto an.

2. Wählen Sie die Option aus, die für Ihre Aufgabe geeignet ist:

So testen Sie Folgendes:	Vorgehensweise:
Eine mit einem Benutzer verknüpfte Richtlinie	Wählen Sie die Option Users (Benutzer) in der Liste Users, Groups, and Roles (Benutzer, Gruppen und Rollen). Wählen Sie dann den Benutzer.
Eine mit einer Gruppe verknüpfte Richtlinie	Wählen Sie die Option Groups (Gruppen) in der Liste Users, Groups, and Roles (Benutzer, Gruppen und Rollen). Wählen Sie dann die Gruppe.
Eine mit einer Rolle verknüpfte Richtlinie	Wählen Sie die Option Roles (Rollen) in der Liste Users, Groups, and Roles (Benutzer, Gruppen und Rollen). Wählen Sie dann die Rolle.
Eine mit einer Ressource verknüpfte Richtlinie	Siehe Step 9.
Eine benutzerdefinierte Richtlinie für einen Benutzer, eine Gruppe oder eine Rolle	Wählen Sie Create New Policy (Neue Richtlinie erstellen). Geben Sie im neuen Bereich Policies (Richtlinien) eine Richtlinie ein oder fügen Sie sie ein und wählen Sie dann Apply (Anwenden) aus.

> **Tipp**
>
> Um eine mit einer Gruppe verknüpfte Richtlinie zu testen, können Sie den IAM-Richtliniensimulator direkt von der IAM-Konsole aus starten: Wählen Sie im Navigationsbereich die Option Groups. Wählen Sie den Namen der Gruppe aus, für die Sie eine Richtlinie testen möchten, und klicken Sie dann auf die Registerkarte Permissions (Berechtigungen). Klicken Sie auf Simulieren.
> So testen Sie eine vom Kunden verwaltete Richtlinie, die mit einem Benutzer verknüpft ist: Wählen Sie im Navigationsbereich die Option Users (Benutzer). Klicken Sie auf den Namen des Benutzers, für den Sie eine Richtlinie testen möchten. Klicken Sie dann auf die Registerkarte Permissions (Berechtigungen) und erweitern Sie die zu testende

Richtlinie. Wählen Sie ganz rechts die Option Simulate policy (Richtlinie simulieren). Der IAM-Richtliniensimulator wird in einem neuen Fenster geöffnet, in dem die ausgewählte Richtlinie im Bereich Policies angezeigt wird.

3. (Optional) Wenn Ihr Konto Mitglied einer Organisation in AWS Organizations ist, dann aktivieren Sie das Kontrollkästchen neben AWS Organizations -SCPs, um SCPs in Ihre simulierte Bewertung einzubeziehen. SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) festlegen. Die SCP beschränkt Berechtigungen für Entitäten in Mitgliedskonten. Wenn eine Service-Kontrollrichtlinie einen Service oder eine Aktion blockiert, kann keine Entität in diesem Konto auf diesen Service zugreifen oder diese Aktion ausführen. Dies gilt auch, wenn ein Administrator diesem Service oder dieser Aktion über eine IAM- oder Ressourcenrichtlinie explizit Berechtigungen erteilt.

   Wenn Ihr Konto kein Mitglied einer Organisation ist, wird das Kontrollkästchen nicht angezeigt.

4. (Optional) Sie können eine Richtlinie testen, die als Berechtigungsgrenze für eine IAM-Entity (Benutzer oder Rolle) festgelegt ist, jedoch nicht für Gruppen. Wenn derzeit eine Berechtigungsgrenzen-Richtlinie für die Entity festgelegt ist, wird sie im Bereich Policies (Richtlinien) angezeigt. Sie können nur eine Berechtigungsgrenze für eine Entity festlegen. Um eine andere Berechtigungsgrenze zu testen, können Sie eine benutzerdefinierte Berechtigungsgrenze erstellen. Wählen Sie dazu die Option Create New Policy (Neue Richtlinie erstellen). Ein neuer Bereich Policies (Richtlinien) wird geöffnet. Wählen Sie im Menü Custom IAM Permissions Boundary Policy (Richtlinie zu benutzerdefinierten IAM-Berechtigungsgrenzen). Geben Sie einen Namen für die neue Richtlinie und anschließend im Feld unten eine Richtlinie ein oder kopieren Sie sie in das Feld. Wählen Sie Apply (Anwenden) aus, um die Richtlinie zu speichern. Wählen Sie als Nächstes Back (Zurück) aus, um zum ursprünglichen Bereich Policies (Richtlinien) zurückzukehren. Aktivieren Sie dann das Kontrollkästchen neben der Berechtigungsgrenze, die Sie für die Simulation verwenden möchten.

5. (Optional) Sie können nur eine Teilmenge von Richtlinien testen, die einem Benutzer, einer Gruppe oder einer Rolle angefügt sind. Deaktivieren Sie dazu im Bereich Policies (Richtlinien) das Kontrollkästchen neben jeder Richtlinie, die Sie ausschließen möchten.

6. Klicken Sie unter Policy Simulator (Richtliniensimulator) auf Select service (Service wählen) und wählen Sie den zu testenden Service. Klicken Sie dann auf Select actions (Aktionen wählen) und wählen Sie eine oder mehrere zu testende Aktionen. Obwohl in den Menüs die verfügbaren Auswahlmöglichkeiten für nur jeweils einen Service aufgeführt werden, werden unter Action Settings and Results (Aktionseinstellungen und Ergebnisse) sämtliche von Ihnen ausgewählte Services und Aktionen angezeigt.

7. (Optional) Wenn eine der in Step 2 und Step 5 ausgewählten Richtlinien Bedingungen mit den AWSglobalen Bedingungsschlüsseln enthält, dann geben Sie Werte für diese Schlüssel an. Erweitern Sie hierzu den Abschnitt Global Settings (Globale Einstellungen) und geben Sie die entsprechenden Werte für die dort angezeigten Schlüsselnamen ein.

   > **Warning**
   >
   > Wenn Sie den Wert für einen Bedingungsschlüssel nicht angeben, wird dieser Schlüssel während der Simulation ignoriert. In einigen Fällen führt dies zu einem Fehler und die Ausführung der Simulation schlägt fehl. In anderen Fällen wird die Simulation ausgeführt, doch die Ergebnisse sind möglicherweise nicht zuverlässig. In diesen Fällen entspricht die Simulation nicht den realen Bedingungen, zu denen ein Wert für den Bedingungsschlüssel oder die Variable gehört.

8. (Optional) Jede ausgewählte Aktion wird in der Liste Action Settings and Results (Aktionseinstellungen und Ergebnisse) mit dem Status Not simulated (Nicht simuliert) in der Spalte Permission (Berechtigung) angezeigt, bis Sie die Simulation tatsächlich ausführen. Bevor Sie die Simulation ausführen, können Sie die jeweilige Aktion mit einer Ressource konfigurieren. Um einzelne Aktionen für ein bestimmtes Szenario zu konfigurieren, klicken Sie auf den Pfeil, um die Zeile der Aktion zu erweitern. Wenn die Aktion Berechtigungen auf Ressourcenebene unterstützt, können Sie den Amazon-Ressourcenname (ARN) der betreffenden Ressource eingeben, deren Zugriff Sie testen möchten. Standardmäßig ist jede Ressource mit einem Platzhalter (*) festgelegt. Sie können auch einen Wert für jeden Bedingungskontextschlüssel angeben. Wie bereits erwähnt, werden Schlüssel mit leeren Werten ignoriert. Dies kann dazu führen, dass die Simulation fehlschlägt oder die Ergebnisse unzuverlässig sind.

   a. Klicken Sie auf den Pfeil neben dem Namen der Aktion, um jede Zeile zu erweitern, und konfigurieren Sie zusätzliche Informationen, die zur genauen Simulation der Aktion in Ihrem Szenario erforderlich sind. Wenn die Aktion Berechtigungen auf Ressourcenebene erfordert, können Sie den Amazon-Ressourcenname (ARN) der bestimmten Ressource eingeben, für die Sie den Zugriff simulieren möchten. Standardmäßig ist jede Ressource mit einem Platzhalter (*) festgelegt.

   b. Wenn die Aktion Berechtigungen auf Ressourcenebene unterstützt, diese jedoch nicht benötigt, können Sie auf Add Resource (Ressource hinzufügen) klicken, um den Ressourcentyp auszuwählen, den Sie zur Simulation hinzufügen möchten.

   c. Wenn eine der ausgewählten Richtlinien ein `Condition`-Element enthält, das auf einen Kontextschlüssel für den Service dieser Aktion verweist, wird dieser Schlüsselname unter

der Aktion angezeigt. Sie können den Wert festlegen, der bei der Simulation dieser Aktion für die angegebene Ressource verwendet werden soll.

Aktionen, die verschiedene Gruppen von Ressourcentypen erfordern

Unter verschiedenen Umständen erfordern einige Aktionen unterschiedliche Ressourcentypen. Jede Gruppe von Ressourcentypen ist mit einem Szenario verknüpft. Wenn eines dieser Szenarien auf Ihre Simulation zutrifft, wählen Sie es. Der Richtliniensimulator benötigt die für das jeweilige Szenario geeigneten Ressourcentypen. In der folgenden Liste werden alle unterstützten Szenariomöglichkeiten sowie die Ressourcen angezeigt, die Sie zum Ausführen der Simulation definieren müssen.

Alle der folgenden Amazon EC2-Szenarien setzen voraus, dass Sie die Ressourcen `instance`, `image` und `security-group` festlegen. Wenn Ihr Szenario ein EBS-Volume enthält, müssen Sie dieses `volume` als Ressource festlegen. Wenn das Amazon EC2-Szenario eine Virtual Private Cloud (VPC) umfasst, müssen Sie die Ressource `network-interface` angeben. Wenn es ein IP-Subnetz umfasst, müssen Sie die Ressource `subnet` angeben. Weitere Informationen zu Amazon EC2-Szenarien finden Sie unter Unterstützte Plattformen im Amazon EC2 Leitfaden.

- EC2-VPC- InstanceStore

  instance, image, security-group, network-interface
- EC2-VPC- -Subnetz InstanceStore

  instance, image, security-group, network-interface, subnet
- EC2-VPC-EBS

  instance, image, security-group, network-interface, volume
- EC2-VPC-EBS-Subnet

  instance, image, security-group, network-interface, subnet, volume

9. (Optional) Wenn Sie eine ressourcenbasierte Richtlinie in Ihre Simulation einbinden möchten, müssen Sie zunächst die Aktionen auswählen, die Sie für diese Ressource in Step 6 simulieren möchten. Erweitern Sie die Zeilen für die ausgewählten Aktionen und geben Sie den ARN der Ressource mit einer zu simulierenden Richtlinie ein. Wählen Sie anschließend die Option Include Resource Policy (Ressourcenrichtlinie einschließen) neben dem Textfeld ARN. Der IAM-Richtliniensimulator unterstützt derzeit nur ressourcenbasierte Richtlinien der

folgenden Services: Amazon S3 (nur ressourcenbasierte Richtlinien, ACLs werden derzeit nicht unterstützt), Amazon SQS, Amazon SNS und entsperrte S3 Glacier-Tresore (gesperrte Tresore werden derzeit nicht unterstützt).

10. Wählen Sie rechts oben die Option Run Simulation (Simulationen ausführen).

    Die Spalte Permission (Berechtigung) in der jeweiligen Zeile der Liste Action Settings and Results (Aktionseinstellungen und Ergebnisse) zeigt das Simulationsergebnis dieser Aktion für die angegebene Ressource an.

11. Um festzustellen, welche Anweisung in einer Richtlinie explizit eine Aktion zulässt oder verweigert, wählen Sie den Link ***N*** matching statement(s) (übereinstimmende Aussage(n)) in der Spalte Permissions (Berechtigungen) aus, um die Zeile zu erweitern. Klicken Sie dann auf den Link Show statement (Anweisung anzeigen). Im Bereich Policies (Richtlinien) wird die relevante Richtlinie mit der Anweisung, die das Simulationsergebnis beeinflusst hat, hervorgehoben angezeigt.

> **Note**
>
> Wenn eine Aktion implizit verweigert wird d. h. wenn die Aktion nur verweigert wird, weil sie nicht explizit zugelassen ist, werden die Optionen List (Anweisung auflisten) und Show statement (Anweisung anzeigen) nicht angezeigt.

Fehlerbehebung bei Meldungen in der Konsole des IAM-Richtliniensimulators

In der folgenden Tabelle werden die Informations- und Warnmeldungen aufgeführt, die bei der Verwendung des IAM-Richtliniensimulators auftreten können. Die Tabelle enthält außerdem Schritte, die Sie zur Fehlerbehebung ausführen können.

Fehlermeldung	Schritte zur Fehlerbehebung
Diese Richtlinie wurde bearbeitet. Änderungen werden nicht in Ihrem Konto gespeichert.	Es ist keine Aktion erforderlich. Diese Meldung dient nur zu Informati onszwecken. Wenn Sie eine vorhandene Richtlinie im IAM-Richtliniensimulator bearbeite n, wirkt sich die Änderung nicht auf Ihr AWS-Konto aus. Die Richtliniensimulator ermöglich

Fehlermeldung	Schritte zur Fehlerbehebung
	t es Ihnen, ausschließlich für Testzwecke Änderungen an Richtlinien vorzunehmen.
Die Ressourcenrichtlinie kann nicht abgerufen werden. Grund: *detaillierte Fehlermeldung*	Der Richtliniensimulator kann nicht auf eine angeforderte ressourcenbasierte Richtlinie zugreifen. Vergewissern Sie sich, dass der angegebene Ressourcen-ARN korrekt ist und dass der Benutzer, der die Simulation ausführt, die Berechtigung zum Lesen der Ressource nrichtlinie besitzt.
Eine oder mehrere Richtlinien benötigen Werte in den Simulationseinstellungen. Ohne diese Werte schlägt die Simulation möglicherweise fehl.	Diese Meldung wird angezeigt, wenn die getestete Richtlinie Bedingungsschlüsse l oder Variablen enthält, Sie jedoch keine Werte für diese Schlüssel oder Variablen unter Simulation Settings (Simulationseinstellungen) angegeben haben. Um diese Meldung zu schließen, wählen Sie Simulation Settings (Simulationseinstellungen) aus und geben einen Wert für den jeweilige n Bedingungsschlüssel oder die jeweilige Variable ein.
Sie haben Richtlinien geändert. Diese Ergebnisse sind nicht mehr gültig.	Diese Meldung wird angezeigt, wenn Sie die ausgewählte Richtlinie geändert haben, während die Ergebnisse im Bereich Results (Ergebnisse) angezeigt wurden. Die im Bereich Results (Ergebnisse) angezeigten Ergebnisse wird nicht dynamisch aktualisiert. Um diese Meldung zu schließen, wählen Sie erneut Run Simulation (Simulation ausführen) aus, um die neuen Simulationsergebnisse basierend auf den im Bereich Policies (Richtlin ien) vorgenommenen Änderungen anzuzeigen.

Fehlermeldung	Schritte zur Fehlerbehebung
Die Ressource, die Sie für diese Simulation eingegeben haben, entspricht nicht diesem Service.	Diese Meldung wird angezeigt, wenn Sie einen Amazon-Ressourcenname (ARN) im Bereich Simulation Settings (Simulationseinstellungen) eingegeben haben, der nicht dem Service entspricht, den Sie für die aktuelle Simulatio n ausgewählt haben. Diese Meldung wird z. B. angezeigt, wenn Sie einen ARN für eine Amazon DynamoDB-Ressource angeben, jedoch Amazon Redshift als zu simulierenden Service auswählen. Führen Sie einen der folgenden Schritte aus, um diese Meldung zu schließen: • Entfernen Sie den ARN aus dem Feld im Bereich Simulation Settings (Simulati onseinstellungen). • Wählen Sie den Service aus, der dem unter Simulation Settings (Simulationseinste llungen) festgelegten ARN entspricht.
Diese Aktion gehört zu einem Service, der zusätzlich zu ressourcenbasierten Richtlini en spezielle Zugriffskontrollmechanismen unterstützt, wie z. B. Amazon S3 ACLs oder S3 Glacier Vault Lock-Richtlinien. Der Richtlini ensimulator bietet keine Unterstützung für diese Mechanismen, sodass sich die Ergebnisse von Ihrer Produktionsumgebung unterscheiden können.	Es ist keine Aktion erforderlich. Diese Meldung dient nur zu Informati onszwecken. In der aktuellen Version bewertet der Richtliniensimulator Richtlinien, die mit Benutzern und Benutzergruppen verknüpft sind, und kann ressourcenbasierte Richtlini en für Amazon S3, Amazon SQS, Amazon SNS und S3 Glacier bewerten. Der Richtlini ensimulator unterstützt nicht alle Zugriffsk ontrollmechanismen, die von anderen AWS - Services unterstützt werden.

Fehlermeldung	Schritte zur Fehlerbehebung
DynamoDB FGAC wird derzeit nicht unterstützt.	Es ist keine Aktion erforderlich. Diese Informationsmeldung bezieht sich auf eine differenzierte Zugriffskontrolle. Eine fein abgestimmte Zugriffskontrolle bietet die Möglichkeit, IAM-Richtlinienbedingungen zu verwenden, um zu bestimmen, wer auf einzelne Datenelemente und Attribute in DynamoDB-Tabellen und -Indizes zugreifen kann. Sie bezieht sich auch auf die Aktionen, die für diese Tabellen und Indizes ausgeführt werden können. Die aktuelle Version des IAM-Richt liniensimulators bietet keine Unterstützung für diesen Typ von Richtlinienbedingungen. Weitere Informationen zur fein abgestuft en DynamoDB-Zugriffskontrolle finden Sie unter Fein abgestufte Zugriffskontrolle für DynamoDB.
Sie verfügen über Richtlinien, die nicht mit der Richtliniensyntax übereinstimmen. Sie können die Richtlinienvalidierung zum Überprüfen und Akzeptieren der empfohlenen Updates für Ihre Richtlinien verwenden.	Diese Meldung wird oben über der Richtlini enliste angezeigt, wenn Sie über Richtlini en verfügen, die nicht die IAM-Richtliniengra mmatik erfüllen. Um diese Richtlinien zu simulieren, lesen Sie die Optionen für die Richtlinienvalidierung unter Validieren von IAM-Richtlinien, um diese Richtlinien zu identifiz ieren und zu beheben.
Diese Richtlinie muss aktualisiert werden, um die neuesten Regeln der Richtliniensyntax zu erfüllen.	Diese Meldung wird angezeigt, wenn Ihre Richtlinien nicht der IAM-Richtliniengrammatik entsprechen. Um diese Richtlinien zu simuliere n, lesen Sie die Optionen für die Richtlini envalidierung unter Validieren von IAM-Richt linien, um diese Richtlinien zu identifizieren und zu beheben.

Verwenden des IAM-Richtliniensimulators (und der API)AWS CLIAWS

Die Befehle des Richtliniensimulators erfordern den Aufruf von API-Operationen für die folgenden zwei Aktionen:

1. Bewerten der Richtlinien und Zurückgeben der Liste der Kontextschlüssel, auf die sie verweisen. Sie müssen wissen, auf welche Kontextschlüssel verwiesen wird, sodass Sie im nächsten Schritt Werte für diese angeben können.
2. Simulieren von Richtlinien, wobei eine Liste der Aktionen, Ressourcen und Kontextschlüssel bereitgestellt wird, die während der Simulation verwendet werden.

Aus Sicherheitsgründen werden die API-Operationen in zwei Gruppen unterteilt:

- API-Operationen, die nur Richtlinien simulieren, die direkt als Zeichenfolgen an die API weitergeleitet werden. Dieses Set beinhaltet GetContextKeysForCustomPolicyund SimulateCustomPolicy.
- API-Operationen, die die Richtlinien simulieren, die dem angegebenen IAM-Benutzer (oder der -Gruppe, -Rolle oder -Ressource) zugeordnet sind. Da diese API-Operationen auch Details zu Berechtigungen offenlegen können, die anderen IAM-Entitys zugewiesen sind, sollten Sie erwägen, den Zugriff auf diese API-Operationen einzuschränken. Dieses Set beinhaltet GetContextKeysForPrincipalPolicyund SimulatePrincipalPolicy. Weitere Informationen zum Einschränken des Zugriffs auf API-Operationen finden Sie unter Beispielrichtlinien: AWS Identity and Access Management (IAM).

In beiden Fällen simulieren die API-Operationen die Auswirkungen einzelner oder mehrerer Richtlinien auf eine Liste der Aktionen und Ressourcen. Die jeweilige Aktion bildet mit der jeweiligen Ressource ein Paar. Die Simulation bestimmt, ob die Richtlinien diese Aktion für diese Ressource zulassen oder verweigern. Sie können auch Werte für Kontextschlüssel angeben, auf die Ihre Richtlinien verweisen. Sie können die Liste der Kontextschlüssel abrufen, auf die die Richtlinien verweisen, indem Sie zuerst `GetContextKeysForCustomPolicy` oder `GetContextKeysForPrincipalPolicy` aufrufen. Wenn Sie keinen Wert für einen Kontextschlüssel angeben, wird die Simulation weiterhin ausgeführt. Die Ergebnisse sind jedoch möglicherweise nicht zuverlässig, da der Richtliniensimulator diesen Kontextschlüssel nicht in der Bewertung berücksichtigen kann.

Um die Liste der Kontextschlüssel (AWS CLI, AWS API) abzurufen

Verwenden Sie die folgenden Informationen, um eine Liste von Richtlinien zu bewerten und eine Liste der Kontextschlüssel zurückzugeben, die in den Richtlinien verwendet werden.

- AWS CLI: `aws iam get-context-keys-for-custom-policy` und `aws iam get-context-keys-for-principal-policy`
- AWS API: `GetContextKeysForCustomPolicy`und `GetContextKeysForPrincipalPolicy`

Um IAM-Richtlinien zu simulieren (AWS CLI, AWS API)

Verwenden Sie die folgenden Informationen zum Simulieren von IAM-Richtlinien, um die effektiven Berechtigungen eines Benutzers zu bestimmen.

- AWS CLI: `aws iam simulate-custom-policy` und `aws iam simulate-principal-policy`
- AWS API: `SimulateCustomPolicy`und `SimulatePrincipalPolicy`

Hinzufügen und Entfernen von IAM-Identitätsberechtigungen

Berechtigungen für eine Identität (Benutzer, Gruppe oder Rolle) werden mithilfe von Richtlinien definiert. Sie können Berechtigungen hinzufügen und entfernen, indem Sie IAM-Richtlinien für eine Identität mithilfe der AWS Management Console, der AWS Command Line Interface (AWS CLI) oder der API anhängen und trennen. AWS Sie können auch Richtlinien verwenden, um Berechtigungsgrenzen nur für Entitäten (Benutzer oder Rollen) festzulegen, die dieselben Methoden verwenden. Berechtigungsgrenzen sind eine erweiterte AWS Funktion, die die maximalen Berechtigungen steuert, die eine Entität haben kann.

Themen

Terminologie

Wenn Sie Berechtigungsrichtlinien mit Identitäten (Benutzer, Gruppen und Rollen) verknüpfen, variieren Terminologie und Verfahren in Abhängigkeit davon, ob Sie mit einer verwalteten oder einer eingebundenen Richtlinie arbeiten:

- Attach (Anfügen) – Wird mit verwalteten Richtlinien verwendet. Sie fügen eine verwaltete Richtlinie an eine Identität (Benutzer, Gruppe oder Rolle) an. Durch Anfügen einer Richtlinie werden die Berechtigungen in der Richtlinie auf die Identität angewendet.
- Detach (Trennen) – Wird mit verwalteten Richtlinien verwendet. Sie trennen eine verwaltete Richtlinie von einer IAM-Identität (Benutzer, Gruppe oder Rolle). Durch Trennen einer Richtlinie werden ihre Berechtigungen von der Identität entfernt.
- Embed (Einbetten) – Wird mit Inline-Richtlinien verwendet. Sie betten eine Inline-Richtlinie in eine Identität (Benutzer, Gruppe oder Rolle) ein. Durch Einbetten einer Richtlinie werden die Berechtigungen in der Richtlinie auf die Identität angewendet. Da eine Inline-Richtlinie in der Identität gespeichert wird, wird sie eingebettet und nicht angefügt, wobei die Ergebnisse ähnlich sind.

  > **Note**
  >
  > Sie können eine eingebundene Richtlinie für eine serviceverknüpfte Rolle nur in den Service einbetten, der von der Rolle abhängt. Informationen dazu, ob diese Funktion von Ihrem Service unterstützt wird, finden Sie in der entsprechenden AWS -Dokumentation.

- Delete (Löschen) – Wird mit Inline-Richtlinien verwendet. Sie löschen eine Inline-Richtlinie für eine IAM-Identität (Benutzer, Gruppe oder Rolle). Durch das Löschen einer Richtlinie werden ihre Berechtigungen von der Identität entfernt.

  > **Note**
  >
  > Sie können eine Inline-Richtlinie für eine serviceverknüpfte Rolle nur aus dem Service löschen, der von der Rolle abhängt. Informationen dazu, ob diese Funktion von Ihrem Service unterstützt wird, finden Sie in der entsprechenden AWS -Dokumentation.

Sie können die Konsole oder die AWS API verwenden AWS CLI, um jede dieser Aktionen auszuführen.

Weitere Informationen

- Weitere Informationen zum Unterschied zwischen verwalteten und eingebundenen Richtlinien finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien.
- Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten.
- Weitere allgemeine Informationen zu IAM-Richtlinien finden Sie unter Berechtigungen und Richtlinien in IAM.
- Weitere Informationen zu IAM-Benutzerrichtlinien finden Sie unter Validieren von IAM-Richtlinien.
- Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Anzeigen der Identitätsaktivitäten

Bevor Sie die Berechtigungen für eine Identität (Benutzer, Gruppe oder Rolle) ändern, sollten Sie deren kürzliche Aktivität auf Serviceebene überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Hinzufügen von IAM-Identitätsberechtigungen (Konsole)

Sie können das verwenden AWS Management Console , um einer Identität (Benutzer, Benutzergruppe oder Rolle) Berechtigungen hinzuzufügen. Fügen Sie dazu verwaltete Richtlinien für die gewünschten Berechtigungen an oder geben Sie eine Richtlinie an, die als Berechtigungsgrenze dient. Sie können auch eine eingebundene Richtlinie einbetten.

So verwenden Sie eine verwaltete Richtlinie als Berechtigungsrichtlinie für eine Identität (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. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien die Optionsschaltfläche neben dem Namen der anzufügenden Richtlinie aus. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie Actions (Aktionen) und dann Attach policy(Richtlinie anfügen).

5. Wählen Sie mindestens eine Identität aus, an die Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Auftraggeber-Entitäten filtern. Nachdem Sie die Identitäten ausgewählt haben, wählen Sie Attach policy (Richtlinie anfügen).

Verwenden einer verwalteten Richtlinie zum Festlegen einer Berechtigungsgrenze (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. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie in der Liste der Richtlinien den Namen der Richtlinie, die Sie festlegen möchten. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie auf der Seite der Richtliniendetails die Registerkarte Entities attached (Angefügte Entitäten) aus und öffnen Sie dann gegebenenfalls den Abschnitt Attached as a permissions boundaries (Als eine Berechtigungsgrenze angefügt) und wählen Sie Set this policy as a permissions boundary (Diese Richtlinie als eine Berechtigungsgrenze festlegen).
5. Wählen Sie einen oder mehrere Benutzer oder Rollen aus, für die die Richtlinie für eine Berechtigungsgrenze verwendet werden soll. Über das Menü Filter und das Suchfeld können Sie die Liste der Auftraggeber-Entitäten filtern. Nachdem Sie die Prinzipale ausgewählt haben, wählen Sie Set permissions boundary (Berechtigungsgrenze festlegen) aus.

So betten Sie eine eingebundene Richtlinie für einen Benutzer oder eine Rolle ein (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 auf Users (Benutzer) oder Roles (Rollen).
3. Wählen Sie in der Liste den Namen des Benutzers oder der Rolle aus, in den bzw. die Sie die Richtlinie einbetten möchten.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wählen Sie Add permissions (Berechtigungen hinzufügen) und dann Create inline policy (Inline-Richtlinie erstellen) aus.

> **Note**
>
> Sie können eingebundene Richtlinien nicht in eine serviceverknüpfte Rolle in IAM einbetten. Da der verknüpfte Service definiert, ob Sie die Berechtigungen der Rolle ändern können, sind Sie möglicherweise in der Lage, zusätzliche Richtlinien von der Service-Konsole, einer API oder der AWS CLI aus hinzuzufügen. Um die serviceverknüpfte Rolle anzuzeigen, lesen Sie unter AWS Dienste, die mit IAM funktionieren nach und wählen Sie Yes (Ja) in der Spalte Service-Linked Role (Serviceverknüpfte Rolle) für den Service.

6. Wählen Sie eine der folgenden Methoden, um die erforderlichen Schritte zum Erstellen der Richtlinie anzuzeigen:

   - Importieren vorhandener verwalteter Richtlinien – Sie können eine verwaltete Richtlinie innerhalb Ihres Kontos importieren und die Richtlinie dann bearbeiten, um sie an Ihre spezifischen Anforderungen anzupassen. Eine verwaltete Richtlinie kann eine AWS verwaltete Richtlinie oder eine vom Kunden verwaltete Richtlinie sein, die Sie zuvor erstellt haben.
   - Erstellen von Richtlinien mit dem visuellen Editor – Sie können eine neue Richtlinie von Grund auf im visuellen Editor erstellen. Wenn Sie den visuellen Editor verwenden, müssen Sie nicht mit der JSON-Syntax vertraut sein.
   - Erstellen von Richtlinien mit dem JSON-Editor – In der Editoroption JSON können Sie eine Richtlinie mithilfe der JSON-Syntax erstellen. Sie können ein neues JSON-Richtliniendokument eingeben oder eine Beispielrichtlinie einfügen.

7. Nachdem Sie eine eingebundene Richtlinie erstellt haben, wird sie automatisch in Ihren Benutzer oder Ihre Rolle eingebettet.

So betten Sie eine eingebundene Richtlinie für eine Gruppe ein (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 auf Benutzergruppen.
3. Wählen Sie in der Liste den Namen der Benutzergruppe, in die eine Richtlinie eingebettet werden soll.
4. Wählen Sie die Registerkarte Berechtigungen, wählen Sie Berechtigungen hinzufügen und wählen Sie dann Inline-Richtlinie erstellen.

5. Führen Sie eine der folgenden Aktionen aus:

   - Wählen Sie die Option Visual aus, um die Richtlinie zu erstellen. Weitere Informationen finden Sie unter Erstellen von Richtlinien mit dem visuellen Editor.
   - Wählen Sie die Option JSON aus, um die Richtlinie zu erstellen.JSON. Weitere Informationen finden Sie unter Erstellen von Richtlinien mit dem JSON-Editor.

6. Wenn Sie mit der Richtlinie zufrieden sind, klicken Sie auf Create policy (Richtlinie erstellen).

So ändern Sie die Berechtigungsgrenze für eine oder mehrere Entitäten (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. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie in der Liste der Richtlinien den Namen der Richtlinie, die Sie festlegen möchten. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie auf der Seite der Richtliniendetails die Registerkarte Entities attached (Angefügte Entitäten) aus und öffnen Sie dann gegebenenfalls den Abschnitt Attached as a permissions boundary (Als eine Berechtigungsgrenze angefügt). Aktivieren Sie das Kontrollkästchen neben den Benutzern oder Rollen, deren Grenzen Sie ändern möchten, und wählen Sie dann Change (Ändern) aus.
5. Wählen Sie eine neue Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll. Sie können über das Suchfeld die Liste der Gruppen filtern. Nachdem Sie die Prinzipale ausgewählt haben, wählen Sie Set permissions boundary (Berechtigungsgrenze festlegen) aus.

Entfernen von IAM-Identitätsberechtigungen (Konsole)

Sie können das verwenden AWS Management Console , um Berechtigungen aus einer Identität (Benutzer, Benutzergruppe oder Rolle) zu entfernen. Trennen Sie hierzu verwaltete Richtlinien, die Berechtigungen steuern, oder entfernen Sie eine Richtlinie an, die als Berechtigungsgrenze dient. Sie können auch eine eingebundene Richtlinie löschen.

Entfernen einer verwalteten Richtlinie, die als Berechtigungsrichtlinie verwendet wurde (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. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien die Optionsschaltfläche neben dem Namen der zu entfernenden Richtlinie aus. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie Aktionen und anschließend Löschen.
5. Wählen Sie die Identitäten aus, von denen Sie die Richtlinie trennen möchten. Sie können über das Suchfeld die Liste der Gruppen filtern. Wählen Sie, nachdem Sie die Identitäten ausgewählt haben, die Option Detach policy (Richtlinie trennen).

Entfernen einer Berechtigungsgrenze (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. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie in der Liste der Richtlinien den Namen der Richtlinie, die Sie festlegen möchten. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie auf der Seite der Richtlinienübersicht die Registerkarte Entities attached (Angefügte Entitäten) aus und öffnen Sie dann gegebenenfalls den Abschnitt Attached as a permissions boundary (Als eine Berechtigungsgrenze angefügt) und wählen Sie die Entitäten aus, aus denen sie die Berechtigungsgrenze entfernen möchten. Wählen Sie dann Remove boundary (Grenze entfernen) aus.
5. Bestätigen Sie, dass Sie die Berechtigungsgrenze entfernen möchten, und wählen Sie Remove boundary (Grenze entfernen) aus.

So löschen Sie eine Inline-Richtlinie (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 auf Groups (Gruppen), Users (Benutzer) oder Roles (Rollen).
3. Wählen Sie in der Liste den Namen der Gruppe, des Benutzers oder der Rolle aus, deren bzw. dessen Richtlinie Sie entfernen möchten.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Aktivieren Sie das Kontrollkästchen neben der Richtlinie und wählen Sie Remove (Entfernen) aus.

6. Wählen Sie im Bestätigungsfeld Remove (Entfernen) aus.

Hinzufügen von IAM-Richtlinien (AWS CLI)

Sie können den verwenden AWS CLI , um einer Identität (Benutzer, Benutzergruppe oder Rolle) Berechtigungen hinzuzufügen. Fügen Sie dazu verwaltete Richtlinien für die gewünschten Berechtigungen an oder geben Sie eine Richtlinie an, die als Berechtigungsgrenze dient. Sie können auch eine eingebundene Richtlinie einbetten.

So verwenden Sie eine verwaltete Richtlinie als Berechtigungsrichtlinie für eine Entität (AWS CLI)

1. (Optional) Um Informationen über eine verwaltete Richtlinie anzuzeigen, führen Sie die folgenden Befehle aus:
   - Auflisten verwalteter Richtlinien: aws iam list-policies
   - Abrufen detaillierter Informationen zu einer verwalteten Richtlinie: get-policy
2. Um eine verwaltete Richtlinie an eine Identität (Benutzer, Benutzergruppe oder Rolle) anzuhängen, verwenden Sie einen der folgenden Befehle:
   - war ich attach-user-policy
   - war ich attach-group-policy
   - war ich attach-role-policy

Verwenden einer verwalteten Richtlinie zum Festlegen einer Berechtigungsgrenze (AWS CLI)

1. (Optional) Um Informationen über eine verwaltete Richtlinie anzuzeigen, führen Sie die folgenden Befehle aus:
   - Auflisten verwalteter Richtlinien: aws iam list-policies
   - Abrufen detaillierter Informationen zu einer verwalteten Richtlinie: aws iam get-policy
2. Um eine verwaltete Richtlinie zum Festlegen der Berechtigungsgrenze für eine Entität (Benutzer oder Rolle) zu verwenden, benutzen Sie einen der folgenden Befehle:
   - war ich put-user-permissions-boundary
   - war ich put-role-permissions-boundary

So betten Sie eine Inline-Richtlinie ein (AWS CLI)

Verwenden Sie zum Einbetten einer Inline-Richtlinie in eine Identität (Benutzer, Gruppe oder Rolle, die keine serviceverknüpfte Rolle ist), einen der folgenden Befehle:

- war ich put-user-policy
- war ich put-group-policy
- war ich put-role-policy

Entfernen von IAM-Richtlinien (AWS CLI)

Sie können die verwenden AWS CLI , um verwaltete Richtlinien, die Berechtigungen kontrollieren, zu trennen oder eine Richtlinie zu entfernen, die als Grenze für Berechtigungen dient. Sie können auch eine eingebundene Richtlinie löschen.

Entfernen einer verwalteten Richtlinie, die als Berechtigungsrichtlinie verwendet wurde (AWS CLI)

1. (Optional) Um Informationen über eine Richtlinie anzuzeigen, führen Sie die folgenden Befehle aus:
   - Auflisten verwalteter Richtlinien: aws iam list-policies
   - Abrufen detaillierter Informationen zu einer verwalteten Richtlinie: aws iam get-policy
2. (Optional) Führen Sie zum Ermitteln von Informationen über die Beziehungen zwischen Richtlinien und Identitäten die folgenden Befehle aus:
   - Listen Sie die Identitäten (Benutzer, Gruppen und Rollen), an die eine verwaltete Richtlinie angefügt ist, wie folgt auf:
     - was ich bin list-entities-for-policy
   - Verwenden Sie zum Auflisten der an eine Identität (Benutzer, Gruppe oder Rolle) angefügten verwalteten Richtlinien einen der folgenden Befehle aus:
     - war ich list-attached-user-policies
     - war ich list-attached-group-policies
     - war ich list-attached-role-policies
3. Verwenden Sie zum Trennen einer verwalteten Richtlinie von einer Identität (Benutzer, Gruppe oder Rolle) einen der folgenden Befehle:
   - war ich detach-user-policy
   - war ich detach-group-policy

- war ich detach-role-policy

Entfernen einer Berechtigungsgrenze (AWS CLI)

1. (Optional) Um anzuzeigen, welche verwaltete Richtlinie zur Zeit verwendet wird, um die Berechtigungsgrenze für einen Benutzer oder eine Rolle festzulegen, führen Sie die folgenden Befehle aus:

   - aws iam get-user
   - aws iam get-role

2. (Optional) Um die Benutzer oder Rollen anzuzeigen, für die eine verwaltete Richtlinie für eine Berechtigungsgrenze verwendet wird, führen Sie den folgenden Befehl aus:

   - war ich list-entities-for-policy

3. (Optional) Um Informationen über eine verwaltete Richtlinie anzuzeigen, führen Sie die folgenden Befehle aus:

   - Auflisten verwalteter Richtlinien: aws iam list-policies
   - Abrufen detaillierter Informationen zu einer verwalteten Richtlinie: aws iam get-policy

4. Um eine Berechtigungsgrenze von einem Benutzer oder einer Rolle zu entfernen, verwenden Sie einen der folgenden Befehle:

   - war ich delete-user-permissions-boundary
   - war ich delete-role-permissions-boundary

So löschen Sie eine Inline-Richtlinie (AWS CLI)

1. (Optional) Verwenden Sie zum Auflisten aller Inline-Richtlinien, die an eine Identität (Benutzer, Gruppe, Rolle) angefügt wurden, einen der folgenden Befehle:

   - war ich list-user-policies
   - war ich list-group-policies
   - war ich list-role-policies

2. (Optional) Verwenden Sie zum Abrufen eines in eine Identität (Benutzer, Gruppe oder Rolle) eingebetteten Inline-Richtliniendokuments einen der folgenden Befehle:

- war ich get-user-policy
- war ich get-group-policy
- war ich get-role-policy

3. Verwenden Sie zum Löschen einer Inline-Richtlinie aus einer Identität (Benutzer, Gruppe oder Rolle, die keine serviceverknüpfte Rolle ist), einen der folgenden Befehle:

   - war ich delete-user-policy
   - war ich delete-group-policy
   - war ich delete-role-policy

Hinzufügen von IAM-Richtlinien (API)AWS

Sie können die AWS API verwenden, um verwaltete Richtlinien anzuhängen, die Berechtigungen steuern, oder eine Richtlinie angeben, die als Berechtigungsgrenze dient. Sie können auch eine eingebundene Richtlinie einbetten.

Um eine verwaltete Richtlinie als Berechtigungsrichtlinie für eine Entität zu verwenden (AWS API)

1. (Optional) Rufen Sie zum Anzeigen von Informationen über eine Richtlinie die folgenden Operationen auf:

   - Um verwaltete Richtlinien aufzulisten: ListPolicies
   - So rufen Sie detaillierte Informationen zu einer verwalteten Richtlinie ab: GetPolicy

2. Um eine verwaltete Richtlinie einer Identität (Benutzer, Benutzergruppe oder Rolle) zuzuordnen, rufen Sie einen der folgenden Vorgänge auf:

   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

So verwenden Sie eine verwaltete Richtlinie, um eine Berechtigungsgrenze (AWS API) festzulegen

1. (Optional) Rufen Sie zum Anzeigen von Informationen über eine verwaltete Richtlinie die folgenden Operationen auf:

   - So listen Sie verwaltete Richtlinien auf: ListPolicies

- So rufen Sie detaillierte Informationen zu einer verwalteten Richtlinie ab: GetPolicy

2. Um eine verwaltete Richtlinie zum Festlegen der Berechtigungsgrenze für eine Entität (Benutzer oder Rolle) zu verwenden, verwenden Sie eine der folgenden Operationen:

   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

Um eine Inline-Richtlinie (AWS API) einzubetten

Rufen Sie zum Einbetten einer Inline-Richtlinie in eine Identität (Benutzer, Gruppe oder Rolle, die keine serviceverknüpfte Rolle ist), eine der folgenden Operationen auf:

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

IAM-Richtlinien (AWS API) werden entfernt

Sie können die AWS API verwenden, um verwaltete Richtlinien zur Steuerung von Berechtigungen zu trennen oder eine Richtlinie zu entfernen, die als Berechtigungsgrenze dient. Sie können auch eine eingebundene Richtlinie löschen.

Um eine verwaltete Richtlinie zu trennen, die als Berechtigungsrichtlinie (AWS API) verwendet wird

1. (Optional) Rufen Sie zum Anzeigen von Informationen über eine Richtlinie die folgenden Operationen auf:

   - So listen Sie verwaltete Richtlinien auf: ListPolicies
   - So rufen Sie detaillierte Informationen zu einer verwalteten Richtlinie ab: GetPolicy

2. (Optional) Rufen Sie zum Ermitteln von Informationen über die Beziehungen zwischen Richtlinien und Identitäten die folgenden Operationen auf:

   - Listen Sie die Identitäten (Benutzer, Gruppen und Rollen), an die eine verwaltete Richtlinie angefügt ist, wie folgt auf:
     - ListEntitiesForPolicy
   - Rufen Sie zum Auflisten der an eine Identität (Benutzer, Gruppe oder Rolle) angefügten verwalteten Richtlinien eine der folgenden Operationen auf:

- ListAttachedUserPolicies
- ListAttachedGroupPolicies
- ListAttachedRolePolicies

3. Rufen Sie zum Trennen einer verwalteten Richtlinie von einer Identität (Benutzer, Gruppe oder Rolle) eine der folgenden Operationen auf:

   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

So entfernen Sie eine Berechtigungsgrenze (AWS API)

1. (Optional) Um anzuzeigen, welche verwaltete Richtlinie zur Zeit verwendet wird, um die Berechtigungsgrenze für einen Benutzer oder eine Rolle festzulegen, verwenden Sie die folgenden Operationen:

   - GetUser
   - GetRole

2. (Optional) Um die Benutzer oder Rollen anzuzeigen, für die eine verwaltete Richtlinie für eine Berechtigungsgrenze verwendet wird, verwenden Sie die folgenden Operationen:

   - ListEntitiesForPolicy

3. (Optional) Rufen Sie zum Anzeigen von Informationen über eine verwaltete Richtlinie die folgenden Operationen auf:

   - Um verwaltete Richtlinien aufzulisten: ListPolicies
   - So rufen Sie detaillierte Informationen zu einer verwalteten Richtlinie ab: GetPolicy

4. Um eine Berechtigungsgrenze von einem Benutzer oder einer Rolle zu entfernen, verwenden Sie die folgenden Operationen:

   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

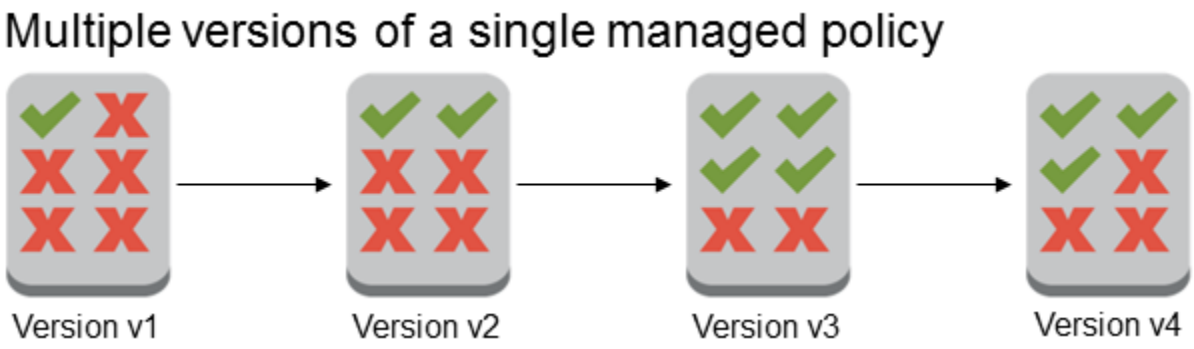
Um eine Inline-Richtlinie (AWS API) zu löschen

1. (Optional) Rufen Sie zum Auflisten aller Inline-Richtlinien, die an eine Identität (Benutzer, Gruppe, Rolle) angefügt sind, eine der folgenden Operationen auf:
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Optional) Rufen Sie zum Abrufen eines in eine Identität (Benutzer, Gruppe oder Rolle) eingebetteten Inline-Richtliniendokuments eine der folgenden Operationen auf:
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy
3. Rufen Sie zum Löschen einer Inline-Richtlinie aus einer Identität (Benutzer, Gruppe oder Rolle, die keine serviceverknüpfte Rolle ist), eine der folgenden Operationen auf:
   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Versioning von IAM-Richtlinien

Wenn Sie Änderungen an einer vom Kunden verwalteten IAM-Richtlinie vornehmen und wenn AWS Sie Änderungen an einer AWS verwalteten Richtlinie vornehmen, überschreibt die geänderte Richtlinie nicht die bestehende Richtlinie. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. IAM speichert bis zu fünf Versionen Ihrer vom Kunden verwalteten Richtlinien. IAM unterstützt Versioning nicht für eingebundene Richtlinien.

Versioning für eine kundenverwaltete Richtlinie wird in der folgenden Abbildung dargestellt. In diesem Beispiel werden die Versionen 1-4 gespeichert. Sie können bis zu fünf verwaltete Richtlinienversionen in IAM speichern lassen. Wenn Sie eine Richtlinie bearbeiten, die eine sechste gespeicherte Version erstellen würde, können Sie wählen, welche ältere Version nicht mehr gespeichert werden soll. Sie können jederzeit auf eine der anderen vier gespeicherten Versionen zurückgreifen.



Eine Richtlinienversion ist nicht mit dem Richtlinienelement `Version` identisch. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Weitere Informationen zum Richtlinienelement `Version` finden Sie unter IAM-JSON-Richtlinienelemente: Version.

Mithilfe von Versionen können Sie Änderungen an verwalteten Richtlinien nachvollziehen. Es kann beispielsweise vorkommen, dass Sie eine Änderung an einer verwalteten Richtlinie vornehmen und diese Änderung dann unerwünschte Auswirkungen hat. In diesem Fall können Sie zu einer vorherigen Version der verwalteten Richtlinie zurückkehren, indem Sie die vorherige Version zur Standardversion machen.

In den folgenden Themen wird erläutert, wie Sie die Versionsverwaltung für verwaltete Richtlinien verwenden können.

Themen

- Berechtigungen für das Festlegen der Standardversion einer Richtlinie
- Festlegen der Standardversion von vom Kunden verwalteten Richtlinien
- Zurücksetzen von Änderungen mithilfe von Versionen
- Versionsbeschränkungen

Berechtigungen für das Festlegen der Standardversion einer Richtlinie

Welche Berechtigungen zum Festlegen der Standardversion einer Richtlinie erforderlich sind, ist von den AWS -API-Operationen für die betreffende Aufgabe abhängig. Sie können mit den API-Operationen `CreatePolicyVersion` oder `SetDefaultPolicyVersion` die Standardversion einer Richtlinie festlegen. Um jemandem die Berechtigung zu erteilen, die Standardversion einer vorhandenen Richtlinie festzulegen, können Sie Zugriff auf die Aktion `iam:CreatePolicyVersion` oder auf die Aktion `iam:SetDefaultPolicyVersion` gewähren.

Die Aktion `iam:CreatePolicyVersion` ermöglicht ihm dann, eine neue Version der Richtlinie zu erstellen und diese Version als Standard festzulegen. Die Aktion `iam:SetDefaultPolicyVersion` ermöglicht ihm, eine vorhandene Version der Richtlinie als Standard festzulegen.

> **Important**
>
> Durch Verweigern der Aktion `iam:SetDefaultPolicyVersion` in der Richtlinie eines Benutzers wird der Benutzer nicht daran gehindert, eine neue Richtlinienversion zu erstellen und diese als Standard festzulegen.

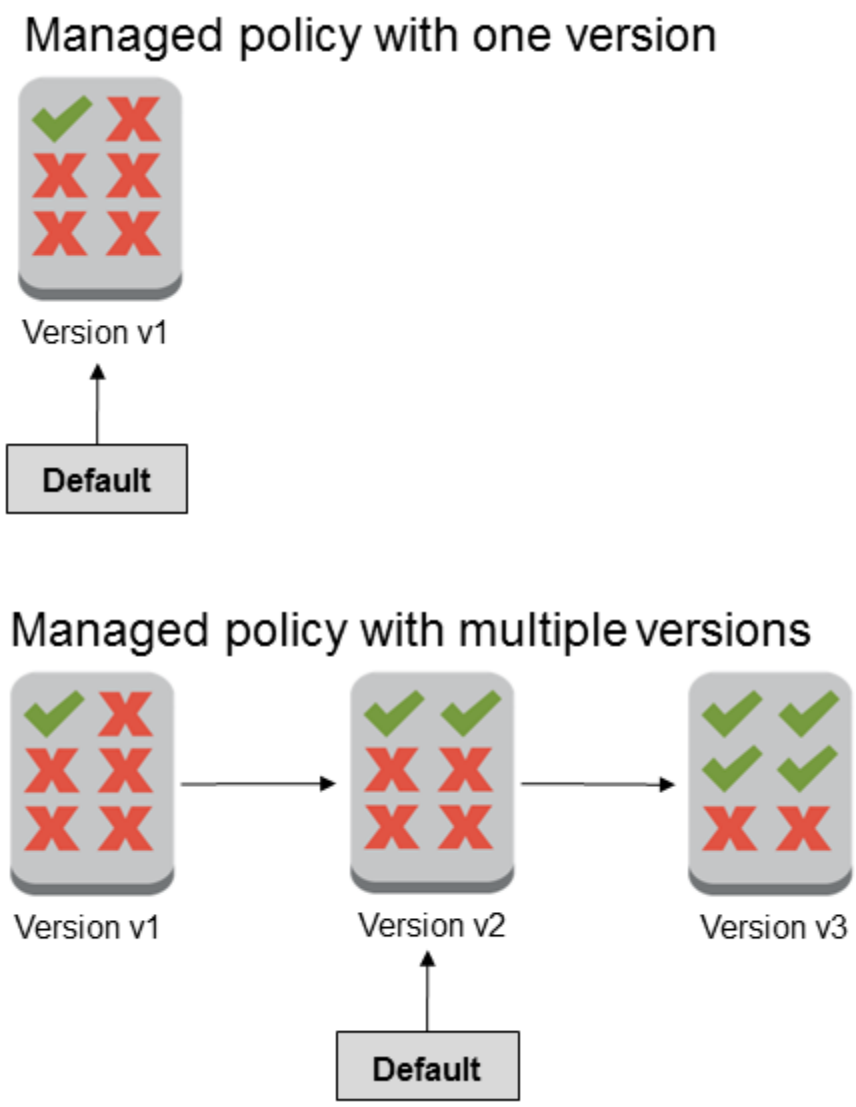
Mit der folgenden Richtlinie können Sie einem Benutzer den Zugriff auf eine vorhandene, kundenverwaltete Richtlinie verweigern, sodass er sie nicht ändern kann:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
}
```

Festlegen der Standardversion von vom Kunden verwalteten Richtlinien

Eine Version einer verwalteten Richtlinie ist immer die Standardversion. Die Standardversion der Richtlinie ist die operative Version, d. h. die Version, die für alle Hauptentitäten (Benutzer, Benutzergruppen und Rollen), denen die verwaltete Richtlinie zugeordnet ist, gültig ist.

Wenn Sie eine kundenverwaltete Richtlinie erstellen, besteht diese zunächst aus nur einer Version, Version 1. Wenn eine verwaltete Richtlinie nur eine Version hat, ist diese automatisch die Standardversion. Bei kundenverwalteten Richtlinien mit mehreren Versionen können Sie festlegen, welche Version als Standardversion verwendet wird. Für AWS verwaltete Richtlinien wird die Standardversion von festgelegt. AWS Das folgende Diagramm verdeutlicht dieses Konzept.



Sie können die Standardversion einer kundenverwalteten Richtlinie so festlegen, dass diese Version auf jede IAM-Identität (Benutzer, Gruppe oder Rolle) angewendet wird, an die die Richtlinie angefügt ist. Sie können die Standardversion nicht für eine AWS verwaltete Richtlinie oder eine Inline-Richtlinie festlegen.

So legen Sie die Standardversion einer kundenverwalteten Richtlinie fest (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. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, für die Sie die Standardversion festlegen möchten. Sie können über das Suchfeld die Liste der Gruppen filtern.

4. Wählen Sie die Registerkarte Policy versions (Richtlinienversionen). Aktivieren Sie das Kontrollkästchen neben der Version, die Sie als Standardversion festlegen möchten, und wählen Sie dann Set as default (Als Standard festlegen).

Informationen dazu, wie Sie die Standardversion einer vom Kunden verwalteten Richtlinie über die AWS Command Line Interface oder die AWS API festlegen, finden Sie unterBearbeiten von vom Kunden verwalteten Richtlinien (AWS CLI).

Zurücksetzen von Änderungen mithilfe von Versionen

Sie können die Standardversion einer kundenverwalteten Richtlinie festlegen, um Ihre Änderungen rückgängig machen zu können. Betrachten wir folgendes Beispielszenario:

Sie erstellen eine kundenverwaltete Richtlinie, mit der Benutzer einen bestimmten Amazon S3-Bucket auf der AWS Management Console verwalten können. Nach dem Erstellen hat die kundenverwaltete Richtlinie nur eine Version, Version 1, die automatisch zur Standardversion wird. Die Richtlinie funktioniert wie vorgesehen.

Später aktualisieren Sie die Richtlinie, um Benutzern die Berechtigung zur Verwaltung eines zweiten Amazon S3-Buckets zu gewähren. IAM erstellt eine neue Version der Richtlinie, Version 2, die die vorgenommenen Änderungen enthält. Sie legen Version v2 als Standard fest, und kurze Zeit später melden Ihre Benutzer, dass sie nicht berechtigt sind, die Amazon S3-Konsole zu verwenden. In diesem Fall kehren Sie zu Version 1 der Richtlinie zurück, da Sie wissen, dass diese wie vorgesehen funktioniert. Dazu legen Sie einfach Version 1 als Standardversion fest. Ihre Benutzer können nun den ursprünglichen Bucket wieder mit der Amazon S3-Konsole verwalten.

Später, nachdem Sie den Fehler in Version v2 der Richtlinie festgestellt haben, aktualisieren Sie die Richtlinie erneut, um die Berechtigung zur Verwaltung des zweiten Amazon S3-Buckets hinzuzufügen. IAM-erstellt eine weitere neue Version der Richtlinie, Version 3. Sie legen Version 3 als Standardversion fest. Diese Version funktioniert nun wie vorgesehen. Jetzt können Sie Version 2 der Richtlinie löschen.

Versionsbeschränkungen

Eine verwaltete -Richtlinie kann bis zu fünf Versionen aufweisen. Wenn Sie Änderungen an einer verwalteten Richtlinie für mehr als fünf Versionen über die AWS Command Line Interface oder die AWS API vornehmen müssen, müssen Sie zunächst eine oder mehrere bestehende Versionen löschen. Wenn Sie die verwenden AWS Management Console, müssen Sie keine Version löschen, bevor Sie Ihre Richtlinie bearbeiten. Beim Speichern der sechsten Version wird dann ein Dialogfeld

angezeigt, in dem Sie aufgefordert werden, mindestens eine nicht standardmäßige Version der Richtlinie zu löschen. Sie können das JSON-Richtliniendokument der einzelnen Versionen anzeigen, um eine geeignete Version auszuwählen. Weitere Informationen zu diesem Dialogfeld finden Sie unter the section called “Bearbeiten von IAM-Richtlinien”.

Sie können abgesehen von der Standardversion beliebige Versionen einer verwalteten Richtlinie löschen. Die Versionsnummern der übrigen Versionen bleiben vom Löschen einer Version unberührt. Daher kann es vorkommen, dass Versionsnummern nicht immer aufeinanderfolgen. Wenn Sie beispielsweise Version 2 und 4 einer verwalteten Richtlinie löschen und zwei neue Versionen erstellen, erhalten Sie im Anschluss die Versionen 1, 3, 5, 6 und 7.

Bearbeiten von IAM-Richtlinien

Eine Richtlinie ist eine Entität, die, angefügt an eine Identität oder Ressource, deren Berechtigungen definiert. Richtlinien werden AWS als JSON-Dokumente gespeichert und als identitätsbasierte Richtlinien in IAM an die Prinzipale angehängt. Sie können eine identitätsbasierte Richtlinie an einen Auftraggeber (oder eine Identität) anfügen – z. B. eine IAM-Gruppe, einen Benutzer oder eine Rolle. Zu den identitätsbasierten Richtlinien gehören AWS verwaltete Richtlinien, vom Kunden verwaltete Richtlinien und Inline-Richtlinien. Sie können vom Kunden verwaltete Richtlinien und Inline-Richtlinien in IAM bearbeiten. AWS verwaltete Richtlinien können nicht bearbeitet werden. Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Themen

- Anzeigen des Richtlinienzugriffs
- Bearbeiten von kundenverwalteten Richtlinien (Konsole)
- Bearbeiten von eingebundenen Richtlinien (Konsole)
- Bearbeiten von vom Kunden verwalteten Richtlinien (AWS CLI)
- Vom Kunden verwaltete Richtlinien (AWS API) bearbeiten

Anzeigen des Richtlinienzugriffs

Bevor Sie die Berechtigungen für eine Richtlinie ändern, sollten Sie deren kürzliche Aktivität auf Serviceebene überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Bearbeiten von kundenverwalteten Richtlinien (Konsole)

Sie können die von Kunden verwalteten Richtlinien bearbeiten, um die in der Richtlinie definierten Berechtigungen zu ändern. Eine kundenverwaltete Richtlinie kann bis zu fünf Versionen aufweisen. Das ist wichtig, denn wenn Sie an einer verwalteten Richtlinie mehr als fünf Versionen Änderungen vornehmen, werden Sie AWS Management Console aufgefordert, zu entscheiden, welche Version gelöscht werden soll. Sie können auch die Standardversion ändern oder eine Version einer Richtlinie löschen, bevor Sie sie bearbeiten, um eine solche Aufforderung zu vermeiden. Weitere Informationen zu Versionen finden Sie unter Versioning von IAM-Richtlinien.

So bearbeiten Sie eine vom Kunden verwaltete Richtlinie (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. Wählen Sie im Navigationsbereich Richtlinien.
3. Wählen Sie in der Richtlinienliste den Namen der zu bearbeitenden Richtlinie. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie die Registerkarte Berechtigungen und anschließend Richtlinie bearbeiten aus.
5. Führen Sie eine der folgenden Aktionen aus:

   - Wählen Sie die Option Visual aus, um Ihre Richtlinie zu ändern, ohne mit der JSON-Syntax vertraut sein zu müssen. Sie können Änderungen am Service, an Aktionen, Ressourcen oder optionalen Bedingungen für jeden Berechtigungsblock in Ihrer Richtlinie vornehmen. Sie können auch eine Richtlinie importieren, um unten in der Richtlinie zusätzliche Berechtigungen hinzuzufügen. Wenn Sie alle gewünschten Änderungen vorgenommen haben, wählen Sie Weiter aus, um fortzufahren.
   - Wählen Sie die Option JSON aus, um Ihre Richtlinie zu ändern, indem Sie Text in das JSON-Textfeld eingeben oder einfügen. Sie können auch eine Richtlinie importieren, um unten in der Richtlinie zusätzliche Berechtigungen hinzuzufügen. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter.

     > **Note**
     >
     > Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen,

strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

6. Überprüfen Sie auf der Seite Review and save (Überprüfen und Speichern) den Bereich Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen) und wählen Sie dann Save changes (Änderungen speichern) aus, um Ihre Arbeit zu speichern.
7. Wenn die verwaltete Richtlinie bereits das Maximum von fünf Versionen aufweist, wird ein Dialogfeld angezeigt, wenn Sie Save changes (Änderungen speichern) auswählen. Damit Ihre neue Version gespeichert wird, wird die älteste Version der Richtlinie, die nicht die Standardversion ist, entfernt und durch diese neue Version ersetzt. Optional können Sie die neue Version als Standardversion der Richtlinie einrichten.

   Wählen Sie Save changes (Änderungen speichern) aus, um Ihre neue Richtlinienversion zu speichern.

So legen Sie die Standardversion einer kundenverwalteten Richtlinie fest (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. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, für die Sie die Standardversion festlegen möchten. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie die Registerkarte Policy versions (Richtlinienversionen). Aktivieren Sie das Kontrollkästchen neben der Version, die Sie als Standardversion festlegen möchten, und wählen Sie dann Set as default (Als Standard festlegen).

So löschen Sie eine Version einer kundenverwalteten Richtlinie (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. Wählen Sie im Navigationsbereich Policies.
3. Wählen Sie den Namen der vom Kunden verwalteten Richtlinie mit der zu löschenden Version. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie die Registerkarte Policy versions (Richtlinienversionen). Aktivieren Sie das Kontrollkästchen neben der Version, die Sie löschen möchten. Wählen Sie dann Delete.

5. Bestätigen Sie, dass Sie die Version löschen möchten, und klicken Sie dann auf Delete (Löschen).

Bearbeiten von eingebundenen Richtlinien (Konsole)

Sie können eine Inline-Richtlinie in der AWS Management Console bearbeiten.

So bearbeiten Sie eine eingebundene Richtlinie für eine Gruppe, einen Benutzer oder eine Rolle (Konsole)

1. Klicken Sie im Navigationsbereich auf Groups (Gruppen), Users (Benutzer) oder Roles (Rollen).
2. Wählen Sie den Namen der Gruppe, des Benutzers oder der Rolle aus, deren bzw. dessen Richtlinie Sie ändern möchten. Wählen Sie dann die Registerkarte Permissions (Berechtigung) und erweitern Sie die Richtlinie.
3. Klicken Sie zum Bearbeiten einer eingebundenen Richtlinie auf Edit Policy (Richtlinie bearbeiten).
4. Führen Sie eine der folgenden Aktionen aus:

   - Wählen Sie die Option Visual aus, um Ihre Richtlinie zu ändern, ohne mit der JSON-Syntax vertraut sein zu müssen. Sie können Änderungen am Service, an Aktionen, Ressourcen oder optionalen Bedingungen für jeden Berechtigungsblock in Ihrer Richtlinie vornehmen. Sie können auch eine Richtlinie importieren, um unten in der Richtlinie zusätzliche Berechtigungen hinzuzufügen. Wenn Sie alle gewünschten Änderungen vorgenommen haben, wählen Sie Weiter aus, um fortzufahren.
   - Wählen Sie die Option JSON aus, um Ihre Richtlinie zu ändern, indem Sie Text in das JSON-Textfeld eingeben oder einfügen. Sie können auch eine Richtlinie importieren, um unten in der Richtlinie zusätzliche Berechtigungen hinzuzufügen. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter. Um die Änderungen zu speichern, ohne dass dies Auswirkungen auf die derzeit angefügten Entitäten hat, deaktivieren Sie das Kontrollkästchen Save as default version (Als Standardversion speichern).

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert

IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

5. Überprüfen Sie auf der Seite Review (Prüfen) die Richtlinienübersicht und wählen Sie dann Save changes (Änderungen speichern) aus, um Ihre Eingaben zu speichern.

Bearbeiten von vom Kunden verwalteten Richtlinien (AWS CLI)

Sie können eine vom Kunden verwaltete Richtlinie über AWS Command Line Interface (AWS CLI) bearbeiten.

> **Note**
>
> Eine verwaltete -Richtlinie kann bis zu fünf Versionen aufweisen. Wenn Sie an einer kundenverwalteten Richtlinie Änderungen vornehmen müssen, sodass mehr als fünf Versionen entstehen, müssen Sie zunächst mindestens eine vorhandene Version löschen.

So bearbeiten Sie eine kundenverwaltete Richtlinie (AWS CLI)

1. (Optional) Um Informationen über eine Richtlinie anzuzeigen, führen Sie die folgenden Befehle aus:
   - Auflisten verwalteter Richtlinien: list-policies
   - Abrufen detaillierter Informationen zu einer verwalteten Richtlinie: get-policy
2. (Optional) Führen Sie zum Ermitteln von Informationen über die Beziehungen zwischen Richtlinien und Identitäten die folgenden Befehle aus:
   - Listen Sie die Identitäten (Benutzer, Gruppen und Rollen), an die eine verwaltete Richtlinie angefügt ist, wie folgt auf:
     - list-entities-for-policy
   - So listen Sie die verwalteten Richtlinien auf, die an eine Identität (Benutzer, Gruppe oder Rolle) angefügt sind:
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Rufen Sie zum Bearbeiten einer vom Kunden verwalteten Richtlinie den folgenden Befehl auf:

- create-policy-version

4. Führen Sie zum Überprüfen einer kundenverwalteten Richtlinie den folgenden IAM Access Analyzer-Befehl auf:

   - validieren-policy

So legen Sie die Standardversion einer vom Kunden verwalteten Richtlinie fest (AWS CLI)

1. (Optional) Rufen Sie zum Auflisten verwalteter Richtlinien den folgenden Befehl auf:

   - list-policies

2. Rufen Sie zum Festlegen der Standardversion einer kundenverwalteten Richtlinie den folgenden Befehl auf:

   - set-default-policy-version

So löschen Sie eine Version einer kundenverwalteten Richtlinie (AWS CLI)

1. (Optional) Rufen Sie zum Auflisten verwalteter Richtlinien den folgenden Befehl auf:

   - list-policies

2. Führen Sie zum Löschen einer kundenverwalteten Richtlinie den folgenden Befehl auf:

   - delete-policy-version

Vom Kunden verwaltete Richtlinien (AWS API) bearbeiten

Sie können eine vom Kunden verwaltete Richtlinie mithilfe der AWS API bearbeiten.

> **Note**
>
> Eine verwaltete -Richtlinie kann bis zu fünf Versionen aufweisen. Wenn Sie an einer kundenverwalteten Richtlinie Änderungen vornehmen müssen, sodass mehr als fünf Versionen entstehen, müssen Sie zunächst mindestens eine vorhandene Version löschen.

Um eine vom Kunden verwaltete Richtlinie (AWS API) zu bearbeiten

1. (Optional) Rufen Sie zum Anzeigen von Informationen über eine Richtlinie die folgenden Operationen auf:
   - Um verwaltete Richtlinien aufzulisten: ListPolicies
   - So rufen Sie detaillierte Informationen zu einer verwalteten Richtlinie ab: GetPolicy
2. (Optional) Rufen Sie zum Ermitteln von Informationen über die Beziehungen zwischen Richtlinien und Identitäten die folgenden Operationen auf:
   - Listen Sie die Identitäten (Benutzer, Gruppen und Rollen), an die eine verwaltete Richtlinie angefügt ist, wie folgt auf:
     - ListEntitiesForPolicy
   - So listen Sie die verwalteten Richtlinien auf, die an eine Identität (Benutzer, Gruppe oder Rolle) angefügt sind:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Rufen Sie zum Bearbeiten einer vom Kunden verwalteten Richtlinie die folgende Operation auf:
   - CreatePolicyVersion
4. Rufen Sie zum Überprüfen einer kundenverwalteten Richtlinie die folgende IAM Access Analyzer-Operation auf:
   - ValidatePolicy

So legen Sie die Standardversion einer vom Kunden verwalteten Richtlinie (AWS API) fest

1. (Optional) Rufen Sie zum Auflisten verwalteter Richtlinien die folgende Operation auf:
   - ListPolicies
2. Rufen Sie zum Festlegen der Standardversion einer kundenverwalteten Richtlinie die folgende Operation auf:
   - SetDefaultPolicyVersion

Um eine Version einer vom Kunden verwalteten Richtlinie (AWS API) zu löschen

1. (Optional) Rufen Sie zum Auflisten verwalteter Richtlinien die folgende Operation auf:
   - ListPolicies
2. Rufen Sie zum Löschen einer kundenverwalteten Richtlinie die folgende Operation auf:
   - DeletePolicyVersion

Löschen von IAM-Richtlinien

Sie können IAM-Richtlinien mit der AWS Management Console, der AWS Command Line Interface (AWS CLI) oder der IAM-API löschen.

> **Note**
>
> Das Löschen von IAM-Richtlinien ist dauerhaft. Nachdem die Richtlinie gelöscht wurde, kann sie nicht wiederhergestellt werden.

Weitere Informationen zum Unterschied zwischen verwalteten und eingebundenen Richtlinien finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien.

Weitere allgemeine Informationen zu IAM-Richtlinien finden Sie unter Berechtigungen und Richtlinien in IAM.

Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

Themen

- Anzeigen des Richtlinienzugriffs
- Erstellen von IAM-Richtlinien (Konsole)
- Löschen von IAM-Richtlinien AWS CLI
- Löschen von IAM-Richtlinien (API)AWS

Anzeigen des Richtlinienzugriffs

Bevor Sie eine Richtlinie löschen, sollten Sie deren kürzliche Aktivität auf Serviceebene überprüfen. Das ist wichtig, da Sie keinem Auftraggeber (Person oder Anwendung) einen noch verwendeten Zugriff entziehen möchten. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Erstellen von IAM-Richtlinien (Konsole)

Sie können eine vom Kunden verwaltete Richtlinie löschen, um sie aus Ihrem AWS-Konto zu entfernen. AWS Verwaltete Richtlinien können nicht gelöscht werden.

So löschen Sie eine kundenverwaltete Richtlinie (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. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie die Optionsschaltfläche neben der vom Kunden verwalteten Richtlinie, die Sie löschen möchten. Sie können über das Suchfeld die Liste der Richtlinien filtern.
4. Wählen Sie Aktionen und anschließend Löschen.
5. Folgen Sie den Anweisungen, um zu bestätigen, dass Sie die Richtlinie löschen möchten, und wählen Sie Delete (Löschen) aus.

So löschen Sie eine eingebundene Richtlinie für eine Gruppe, einen Benutzer oder eine Rolle (Konsole)

1. Klicken Sie im Navigationsbereich auf Groups (Gruppen), Users (Benutzer) oder Roles (Rollen).
2. Wählen Sie den Namen der Gruppe, der Rolle oder des Benutzers aus, deren bzw. dessen Richtlinie Sie löschen möchten. Klicken Sie dann auf die Registerkarte Permissions (Berechtigungen).
3. Aktivieren Sie die Kontrollkästchen neben den Richtlinien, die Sie entfernen möchten, und wählen Sie Remove (Entfernen) aus. Um eine Inline-Richtlinie unter Users (Benutzer) oder Roles (Rollen) zu löschen, wählen Sie Remove (Entfernen) aus, um den Löschvorgang zu bestätigen. Wenn Sie eine einzelne Inline-Richtlinie in Benutzergruppen löschen, geben Sie den Namen der Richtlinie ein und wählen Sie Löschen. Wenn Sie mehrere Inline-Richtlinien in Benutzergruppen

löschen, geben Sie die Anzahl der zu löschenden Richtlinien gefolgt von **`inline policies`** ein und wählen Löschen. Wenn Sie beispielsweise drei Inline-Richtlinien löschen wollen, geben Sie **`3 inline policies`** ein.

Löschen von IAM-Richtlinien AWS CLI

Sie können eine kundenverwaltete Richtlinie mit der AWS Command Line Interface löschen.

So löschen Sie eine vom Kunden verwaltete Richtlinie (AWS CLI)

1. (Optional) Um Informationen über eine Richtlinie anzuzeigen, führen Sie die folgenden Befehle aus:
   - Auflisten verwalteter Richtlinien: list-policies
   - Abrufen detaillierter Informationen zu einer verwalteten Richtlinie: get-policy
2. (Optional) Führen Sie zum Ermitteln von Informationen über die Beziehungen zwischen Richtlinien und Identitäten die folgenden Befehle aus:
   - Führen Sie zum Auflisten der Identitäten (Benutzer, Gruppen und Rollen), an die eine verwaltete Richtlinie angefügt ist, den folgenden Befehl aus:
     - list-entities-for-policy
   - Führen Sie zum Auflisten der an eine Identität (Benutzer, Gruppe oder Rolle) angefügten verwalteten Richtlinien einen der folgenden Befehle aus:
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Führen Sie zum Löschen einer kundenverwalteten Richtlinie den folgenden Befehl auf:
   - delete-policy

So löschen Sie eine Inline-Richtlinie (AWS CLI)

1. (Optional) Verwenden Sie zum Auflisten aller Inline-Richtlinien, die an eine Identität (Benutzer, Gruppe, Rolle) angefügt wurden, einen der folgenden Befehle:
   - war ich list-user-policies
   - war ich list-group-policies

- war ich list-role-policies

2. (Optional) Verwenden Sie zum Abrufen eines in eine Identität (Benutzer, Gruppe oder Rolle) eingebetteten Inline-Richtliniendokuments einen der folgenden Befehle:

   - war ich get-user-policy
   - war ich get-group-policy
   - war ich get-role-policy

3. Verwenden Sie zum Löschen einer Inline-Richtlinie aus einer Identität (Benutzer, Gruppe oder Rolle, die keine serviceverknüpfte Rolle ist), einen der folgenden Befehle:

   - war ich delete-user-policy
   - war ich delete-group-policy
   - war ich delete-role-policy

Löschen von IAM-Richtlinien (API)AWS

Sie können eine vom Kunden verwaltete Richtlinie mithilfe der AWS API löschen.

Um eine vom Kunden verwaltete Richtlinie (AWS API) zu löschen

1. (Optional) Rufen Sie zum Anzeigen von Informationen über eine Richtlinie die folgenden Operationen auf:

   - Um verwaltete Richtlinien aufzulisten: ListPolicies
   - So rufen Sie detaillierte Informationen zu einer verwalteten Richtlinie ab: GetPolicy

2. (Optional) Rufen Sie zum Ermitteln von Informationen über die Beziehungen zwischen Richtlinien und Identitäten die folgenden Operationen auf:

   - Rufen Sie zum Auflisten der Identitäten (Benutzer, Gruppen und Rollen), an die eine verwaltete Richtlinie angefügt ist, die folgende Operation auf:
     - ListEntitiesForPolicy
   - Rufen Sie zum Auflisten der an eine Identität (Benutzer, Gruppe oder Rolle) angefügten verwalteten Richtlinien eine der folgenden Operationen auf:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies

3. Rufen Sie zum Löschen einer kundenverwalteten Richtlinie die folgende Operation auf:

   - DeletePolicy

Um eine Inline-Richtlinie (AWS API) zu löschen

1. (Optional) Rufen Sie zum Auflisten aller Inline-Richtlinien, die an eine Identität (Benutzer, Gruppe, Rolle) angefügt sind, eine der folgenden Operationen auf:

   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies

2. (Optional) Rufen Sie zum Abrufen eines in eine Identität (Benutzer, Gruppe oder Rolle) eingebetteten Inline-Richtliniendokuments eine der folgenden Operationen auf:

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

3. Rufen Sie zum Löschen einer Inline-Richtlinie aus einer Identität (Benutzer, Gruppe oder Rolle, die keine serviceverknüpfte Rolle ist), eine der folgenden Operationen auf:

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde

Als Administrator können Sie Berechtigungen für IAM-Ressourcen (Rollen, Benutzer, Benutzergruppen oder Richtlinien) gewähren, die über ihre Anforderungen hinausgehen. IAM bietet Informationen zum jeweils letzten Zugriff auf diese Informationen, anhand derer Sie nicht verwendete Berechtigungen identifizieren und diese dann entfernen können. Sie können die Informationen zum letzten Zugriff verwenden, um Ihre Richtlinien zu verfeinern und den Zugriff nur auf Services und Aktionen zu gewähren, die Ihre IAM-Identitäten und -Richtlinien verwenden. Auf diese Weise können Sie besser die bewährten Methoden der geringsten Rechte befolgen. Sie können Informationen

zum letzten Zugriff für Identitäten oder Richtlinien anzeigen, die in IAM oder AWS Organizations vorhanden sind.

Sie können die zuletzt abgerufenen Informationen mit Analysatoren für ungenutzten Zugriff kontinuierlich überwachen. Weitere Informationen finden Sie unter Ergebnisse für externen und ungenutzten Zugriff.

Themen

Informationstypen zum letzten Zugriff für IAM

Sie können zwei Arten von Informationen über die letzten Zugriffe auf IAM-Identitäten anzeigen: Informationen über genehmigte AWS -Services und Informationen über genehmigte Aktionen. Zu den Informationen gehören Datum und Uhrzeit des Versuchs, auf eine AWS API zuzugreifen. Bei Aktionen werden die zuletzt aufgerufenen Informationen über Service-Verwaltungsaktionen gemeldet. Managementaktionen umfassen Erstellungs-, Lösch- und Änderungsaktionen. Weitere Informationen zum Anzeigen der Informationen zum letzten Zugriff für IAM finden Sie unter Anzeigen von Informationen zum letzten Zugriff für IAM.

Beispielszenarien zur Verwendung der Informationen zum letzten Zugriff für Entscheidungen zu den Berechtigungen für Ihre IAM-Identitäten finden Sie unter Beispielszenarien für die Verwendung von Informationen zum letzten Zugriff.

Weitere Informationen dazu, wie die Informationen für Managementaktionen bereitgestellt werden, finden Sie unter Wissenswertes über die Informationen zum letzten Zugriff.

Informationen zum letzten Zugriff für AWS Organizations

Wenn Sie sich mit den Anmeldeinformationen für das Verwaltungskonto anmelden, können Sie die Informationen zum letzten Zugriff auf den Dienst für eine AWS Organizations Entität oder Richtlinie in Ihrer Organisation einsehen. AWS Organizations Zu den Entitäten gehören der Organisationsstamm, die Organisationseinheiten (OUs) oder die Konten. Zu den Informationen für, auf die zuletzt zugegriffen wurde, AWS Organizations gehören Informationen über Dienste, die gemäß einer Service Control Policy (SCP) zulässig sind. Die Informationen geben an, welche Prinzipale (Root-Benutzer, IAM-Benutzer oder Rolle) in einer Organisation oder einem Konto zuletzt versucht haben, auf den Service zuzugreifen und wann. Weitere Informationen zum Bericht und zum Anzeigen der zuletzt aufgerufenen Informationen finden Sie AWS Organizations unterSo zeigen Sie Informationen zum letzten Zugriff für Organizations an.

Beispielszenarien zur Verwendung der Informationen zum letzten Zugriff für Entscheidungen zu den Berechtigungen für Ihre Organizations-Entitäten finden Sie unter Beispielszenarien für die Verwendung von Informationen zum letzten Zugriff.

Wissenswertes über die Informationen zum letzten Zugriff

Bevor Sie anhand der Informationen zum letzten Zugriff aus einem Bericht die Berechtigungen für eine IAM-Identität oder -Entität ändern, überprüfen Sie die folgenden Details zu den Informationen.

- Nachverfolgungszeitraum – Die kürzlich erfolgten Aktivitäten werden innerhalb von vier Stunden in der IAM-Konsole angezeigt. Der Nachverfolgungszeitraum für Service-Informationen beträgt mindestens 400 Tage, je nachdem, wann der Service mit der Nachverfolgung von Aktionsinformationen begonnen hat. Der Nachverfolgungszeitraum für Amazon S3 Aktionsinformationen begann am 12. April 2020. Der Nachverfolgungszeitraum für Amazon EC2, IAM- und Lambda-Aktionen begann am 7. April 2021. Der Nachverfolgungszeitraum für alle anderen Services begann am 23. Mai 2023. Eine Liste der Services, für die Informationen über die zuletzt aufgerufene Aktion verfügbar sind, finden Sie unter IAM-Aktion, zuletzt aufgerufene Informationsservices und Aktionen. Weitere Informationen darüber, in welchen Regionen die Aktion zuletzt aufgerufen wurde, finden Sie unter Wo werden die AWS zuletzt aufgerufenen Informationen aufgezeichnet.
- Gemeldete Versuche — Die Daten, auf die der Dienst zuletzt zugegriffen hat, umfassen alle Versuche, auf eine AWS API zuzugreifen, nicht nur die erfolgreichen Versuche. Dazu gehören alle Versuche, die mithilfe der AWS Management Console, der AWS API, eines der SDKs oder eines der Befehlszeilentools unternommen wurden. Ein unerwarteter Eintrag als letztes Service-Zugriffsdatum bedeutet nicht, dass Ihr Konto beschädigt oder infiziert worden ist, da der

Zugriff möglicherweise verweigert wurde. Informationen zu allen API-Aufrufen und ob der Zugriff erfolgreich war oder verweigert wurde, finden Sie in Ihren CloudTrail Protokollen als maßgebliche Quelle.

- PassRole— Die `iam:PassRole` Aktion wird nicht nachverfolgt und ist nicht in den Informationen zum letzten Zugriff auf die IAM-Aktion enthalten.
- Informationen zur Aktion, auf die zuletzt zugegriffen wurde – Informationen zur Aktion, auf die zuletzt zugegriffen wurde, sind für Service-Management-Aktionen verfügbar, auf die IAM-Identitäten zugreifen. Hier finden Sie die Liste der Services und deren Aktionen, für die zuletzt auf Berichtsinformationen zugegriffen wurde.

> **Note**
>
> Informationen zur Aktion, auf die zuletzt zugegriffen wurde, sind für Amazon-S3-Datenaktionen verfügbar.

- Verwaltungsereignisse — IAM stellt Aktionsinformationen für Servicemanagement-Ereignisse bereit, die von protokolliert werden. CloudTrail Manchmal werden CloudTrail Verwaltungsereignisse auch als Operationen auf Kontrollebene oder Ereignisse auf Kontrollebene bezeichnet. Verwaltungsereignisse bieten Einblick in Verwaltungsvorgänge, die an Ressourcen in Ihrem Unternehmen ausgeführt werden AWS-Konto. Weitere Informationen zu Verwaltungsereignissen finden Sie unter Protokollieren von Verwaltungsereignissen im AWS CloudTrail Benutzerhandbuch. CloudTrail
- Berichtsbesitzer – Nur der Auftraggeber, der einen Bericht generiert, kann die Berichtsdetails anzeigen. Das bedeutet, dass Sie beim Anzeigen der Informationen in der möglicherweise warten müssen AWS Management Console, bis sie generiert und geladen sind. Wenn Sie die AWS API AWS CLI oder verwenden, um Berichtsdetails abzurufen, müssen Ihre Anmeldeinformationen mit den Anmeldeinformationen des Prinzipals übereinstimmen, der den Bericht generiert hat. Wenn Sie temporäre Anmeldeinformationen für eine Rolle oder einen Verbundbenutzer verwenden, müssen Sie den Bericht während derselben Sitzung generieren und abrufen. Weitere Hinweise zu Sitzungs-Auftraggeber für angenommene Rollen finden Sie unter AWS JSON-Richtlinienelemente: Principal.
- IAM-Ressourcen – Zu den zuletzt abgerufenen Informationen für IAM gehören IAM-Ressourcen (Rollen, Benutzer, Benutzergruppen und Richtlinien) in Ihrem Konto. Zu den zuletzt aufgerufenen Informationen für Organizations gehören Prinzipale (IAM-Benutzer, IAM-Rollen oder die Root-Benutzer des AWS-Kontos) in der angegebenen Organisationsentität. Die zuletzt zugegriffenen Informationen beinhalten keine nicht authentifizierten Versuche.

- IAM-Richtlinientypen – Die zuletzt aufgerufenen Informationen für IAM umfassen Services, die durch die Richtlinien einer IAM-Identität zugelassen sind. Diese Richtlinien sind einer Rolle oder einem Benutzer direkt oder über eine Gruppe angefügt. Zugriff, der über andere Richtlinientypen gewährt wird, ist in diesem Bericht nicht enthalten. Ausgeschlossen sind die Richtlinientypen ressourcenbasierte Richtlinien, Zugriffskontrolllisten, AWS Organizations -SCPs, IAM-Berechtigungsgrenzen und Sitzungsrichtlinien. Berechtigungen, die von serviceverknüpften Rollen bereitgestellt werden, werden von dem Service definiert, mit dem sie verknüpft sind, und können in IAM nicht geändert werden. Weitere Informationen zu serviceverknüpften Rollen finden Sie unter Verwenden von serviceverknüpften Rollen. Informationen dazu, wie die verschiedenen Richtlinientypen ausgewertet werden, um den Zugriff zuzulassen oder zu verweigern, finden Sie unter Auswertungslogik für Richtlinien.
- Organisations-Richtlinientypen – Die Informationen für AWS Organizations beinhalten nur die Services, die aufgrund der übernommenen Service-Kontrollrichtlinien (SCPs) einer Organisations-Entität zulässig sind. SCPs sind Richtlinien, die an einen Stamm, eine Organisationseinheit oder ein Konto angefügt sind Zugriff, der über andere Richtlinientypen gewährt wird, ist in diesem Bericht nicht enthalten. Ausgeschlossen sind die Richtlinientypen identitätsbasierte Richtlinien, ressourcenbasierte Richtlinien Zugriffskontrolllisten, IAM-Berechtigungsgrenzen und Sitzungsrichtlinien. Informationen dazu, wie die verschiedenen Richtlinientypen ausgewertet werden, um Zugriff zu erlauben oder zu verweigern, finden Sie unter Auswertungslogik für Richtlinien.
- Angabe einer Richtlinien-ID — Wenn Sie die AWS API AWS CLI oder verwenden, um einen Bericht für Informationen zu generieren, auf die in Organizations zuletzt zugegriffen wurde, können Sie optional eine Richtlinien-ID angeben. Der resultierende Bericht enthält Daten für die Services, die nur aufgrund dieser Richtlinie zulässig sind. Die Daten beinhalten die letzten Kontoaktivitäten in der angegebenen Organizations-Entität oder den untergeordneten Elementen der Entität. Weitere Informationen finden Sie unter aws iam generate-organizations-access-report oder GenerateOrganizationsAccessReport.
- Verwaltungskonto der Organisation – Sie müssen sich bei dem Verwaltungskonto Ihrer Organisation anmelden, um die zuletzt abgerufenen Informationen zum Dienst anzuzeigen. Sie können wählen, ob Sie Informationen für das Verwaltungskonto mithilfe der IAM-Konsole AWS CLI, der oder der AWS API anzeigen möchten. Der resultierende Bericht listet alle AWS Dienste auf, da das Verwaltungskonto nicht durch SCPs begrenzt ist. Wenn Sie in der CLI oder API eine Richtlinien-ID angeben, wird die Richtlinie ignoriert. Für jeden Service beinhaltet der Bericht nur Daten für das Hauptkonto. Die Berichte für andere Organisationseinheiten liefern jedoch keine Informationen über die Aktivitäten auf dem Verwaltungskonto.

- Einstellungen für Organisationen – Ein Administrator muss SCPs in Ihrem Organisationsstamm aktivieren, bevor Sie Daten für Organisationen generieren können.

Erforderliche Berechtigungen

Um die zuletzt aufgerufenen Informationen in der anzeigen zu können AWS Management Console, benötigen Sie eine Richtlinie, die die erforderlichen Berechtigungen gewährt.

Berechtigungen für IAM-Informationen

Sie müssen Sie über eine Richtlinie verfügen, die die folgenden Aktionen beinhaltet, um mithilfe der -Konsole die Informationen zum letzten Zugriff für einen IAM-Benutzer, eine Rolle oder Richtlinie anzeigen zu lassen:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam:List*`

Diese Berechtigungen erlauben einem Benutzer, folgende Informationen anzuzeigen:

- Welche Benutzer, Gruppen oder Rollen einer verwalteten Richtlinie angefügt sind
- Auf welche Services ein Benutzer oder eine Rolle zugreifen kann
- Das Datum des letzten Zugriffs auf den Service
- Das letzte Mal, als sie versucht haben, eine bestimmte Amazon EC2-, IAM-, Lambda- oder Amazon S3-Aktion zu verwenden

Um die AWS API AWS CLI oder zum Anzeigen der zuletzt aufgerufenen Informationen für IAM verwenden zu können, benötigen Sie Berechtigungen, die dem Vorgang entsprechen, den Sie verwenden möchten:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Anzeige von IAM-Informationen zum letzten Zugriff erlaubt. Darüber hinaus erlaubt es den Nur-Lese-Zugriff auf das gesamte IAM. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateServiceLastAccessedDetails",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
```

Berechtigungen für AWS Organizations -Informationen

Um die IAM-Konsole zu verwenden, um einen Bericht für die Stamm-, Organisationseinheit- oder Kontoentitäten in Organisationen anzuzeigen, müssen Sie über eine Richtlinie verfügen, die die folgenden Aktionen umfasst:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Um die AWS API AWS CLI oder verwenden zu können, um Informationen über den Dienst, auf den zuletzt zugegriffen wurde, für Organizations anzuzeigen, benötigen Sie eine Richtlinie, die die folgenden Aktionen umfasst:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die die Anzeige von Informationen über den letzten Zugriff auf Services für Organisationen erlaubt. Darüber hinaus ermöglicht es schreibgeschützten Zugriff auf sämtliche Organizations. Diese Richtlinie definiert Berechtigungen für den programmgesteuerten Zugriff und den Konsolenzugriff.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

Sie können auch den OrganizationsPolicyId Bedingungsschlüssel iam: verwenden, um die Generierung eines Berichts nur für eine bestimmte Organisationsrichtlinie zu ermöglichen. Eine Beispielrichtlinie finden Sie unter IAM: Anzeigen von Informationen zum letzten Zugriff auf einen Service für eine Organizations-Richtlinie.

Fehlerbehebungsaktivitäten für IAM und Organizationsentität

In einigen Fällen ist Ihre AWS Management Console letzte Informationstabelle, auf die Sie zugegriffen haben, möglicherweise leer. Oder vielleicht gibt Ihre Anfrage AWS CLI oder Ihre AWS API-Anfrage einen leeren Satz von Informationen oder ein Nullfeld zurück. Überprüfen Sie in solch einem Fall die folgenden Punkte:

- Bei Informationen zum letzten Zugriff auf Aktionen kann es sein, dass eine Aktion, die Sie in der Liste erwarten, dort nicht angezeigt wird. Dies kann entweder passieren, weil die IAM-Identität keine Berechtigungen für die Aktion besitzt, oder weil die Aktion für die zuletzt aufgerufenen Informationen noch AWS nicht verfolgt wird.
- Stellen Sie bei IAM-Benutzern sicher, dass der Benutzer über mindestens eine eingebundene oder verwaltete Richtlinie verfügt (entweder direkt oder über eine Gruppenmitgliedschaft).
- Stellen Sie bei IAM-Gruppen sicher, dass die Gruppe über mindestens eine eingebundene oder verwaltete Richtlinie verfügt.
- Für eine IAM-Gruppe gibt der Bericht Informationen zum letzten Servicezugriff nur für Mitglieder zurück, die mithilfe der Gruppenrichtlinien auf einen Service zugegriffen haben. Wenn Sie wissen möchten, ob ein Mitglied andere Richtlinien verwendet hat, überprüfen Sie die Informationen zum letzten Zugriff für diesen Benutzer.
- Stellen Sie bei IAM-Rollen sicher, dass die Rolle über mindestens eine eingebundene oder verwaltete Richtlinie verfügt.
- Überprüfen Sie für IAM Entitys (Benutzer oder Rollen) andere Richtlinientypen, die sich auf die Berechtigungen dieser Entität auswirken könnten. Dazu gehören ressourcenbasierte Richtlinien, Zugriffskontrolllisten, AWS Organizations Richtlinien, IAM-Berechtigungsgrenzen oder Sitzungsrichtlinien. Weitere Informationen finden Sie unter Richtlinientypen oder Auswerten von Richtlinien in einem einzelnen Konto.
- Stellen Sie bei IAM-Richtlinien sicher, dass die angegebene verwaltete Richtlinie an mindestens einen Benutzer, eine Gruppe mit Mitgliedern oder eine Rolle angefügt ist.
- Vergewissern Sie sich, dass Sie für eine Organisationsentität (Stamm, OU oder Konto) mit den Anmeldeinformationen des Verwaltungskontos Organisationen signiert sind.
- Überprüfen Sie, dass die SCPs in Ihrem Organisationsstamm aktiviert sind.
- Die Information über den letzten Zugriff auf eine Aktion ist nur für die unter IAM-Aktion, zuletzt aufgerufene Informationsservices und Aktionen aufgeführten Aktionen verfügbar.

Wenn Sie Änderungen vornehmen, warten Sie mindestens vier Stunden bis die Aktivitäten in Ihrer IAM-Konsole angezeigt werden. Wenn Sie die AWS API AWS CLI oder verwenden, müssen Sie einen neuen Bericht erstellen, um die aktualisierten Informationen anzuzeigen.

Wo werden die AWS zuletzt aufgerufenen Informationen aufgezeichnet

AWS sammelt Informationen, auf die zuletzt zugegriffen wurde, für die AWS Standardregionen. Wenn weitere Regionen AWS hinzugefügt werden, werden diese Regionen der folgenden Tabelle hinzugefügt, einschließlich des Datums, AWS an dem mit der Erfassung von Informationen in jeder Region begonnen wurde.

- Serviceinformationen – Der Nachverfolgungszeitraum für Services beträgt mindestens 400 Tage oder weniger, wenn Ihre Region innerhalb der letzten 400 Tage mit der Verfolgung dieses Features begonnen hat.
- Aktionsinformationen – Der Nachverfolgungszeitraum für Amazon S3-Managementaktionen begann am 12. April 2020. Der Nachverfolgungszeitraum für die Verwaltungsmaßnahmen von Amazon EC2, IAM und Lambda begann am 7. April 2021. Der Nachverfolgungszeitraum für Verwaltungsmaßnahmen für alle anderen Services begann am 23. Mai 2023. Wenn das Nachverfolgungsdatum einer Region nach dem 23. Mai 2023 liegt, beginnt die Aktion mit den zuletzt aufgerufenen Informationen aus dieser Region zu dem späteren Datum.

Name der Region	Region	Startdatum für Verfolgung
USA Ost (Ohio)	us-east-2	27. Oktober 2017
USA Ost (Nord-Virginia)	us-east-1	1. Oktober 2015
USA West (Nordkalifornien)	us-west-1	1. Oktober 2015
USA West (Oregon)	us-west-2	1. Oktober 2015
Afrika (Kapstadt)	af-south-1	22. April 2020
Asien-Pazifik (Hongkong)	ap-east-1	24. April 2019
Asien-Pazifik (Hyderabad)	ap-south-2	22. November 2022
Asien-Pazifik (Jakarta)	ap-southeast-3	13. Dezember 2021

Name der Region	Region	Startdatum für Verfolgung
Asien-Pazifik (Melbourne)	ap-southeast-4	23. Januar 2023
Asien-Pazifik (Mumbai)	ap-south-1	27. Juni 2016
Asien-Pazifik (Osaka)	ap-northeast-3	11. Februar 2018
Asien-Pazifik (Seoul)	ap-northeast-2	6. Januar 2016
Asien-Pazifik (Singapur)	ap-southeast-1	1. Oktober 2015
Asien-Pazifik (Sydney)	ap-southeast-2	1. Oktober 2015
Asien-Pazifik (Tokio)	ap-northeast-1	1. Oktober 2015
Kanada (Zentral)	ca-central-1	28. Oktober 2017
Europa (Frankfurt)	eu-central-1	1. Oktober 2015
Europa (Irland)	eu-west-1	1. Oktober 2015
Europa (London)	eu-west-2	28. Oktober 2017
Europa (Mailand)	eu-south-1	28. April 2020
Europa (Paris)	eu-west-3	18. Dezember 2017
Europa (Spanien)	eu-south-2	15. November 2022
Europa (Stockholm)	eu-north-1	12. Dezember 2018
Europa (Zürich)	eu-central-2	08. November 2022
Israel (Tel Aviv)	il-central-1	1. August 2023
Naher Osten (Bahrain)	me-south-1	29. Juli 2019
Naher Osten (VAE)	me-central-1	30. August 2022
Südamerika (São Paulo)	sa-east-1	11. Dezember 2015

Name der Region	Region	Startdatum für Verfolgung
AWS GovCloud (US-Ost)	us-gov-east-1	1. Juli 2023
AWS GovCloud (US-West)	us-gov-west-1	1. Juli 2023

Wenn eine Region in der vorherigen Tabelle nicht aufgeführt ist, stehen Informationen zum letzten Servicezugriff für diese Region noch nicht zur Verfügung.

Eine AWS Region ist eine Sammlung von AWS Ressourcen in einem geografischen Gebiet. Regionen werden in Partitionen gruppiert. Die Standardregionen sind die Regionen, die zur `aws`-Partition gehören. Weitere Informationen zu den verschiedenen Partitionen finden Sie unter Amazon-Ressourcennamen (ARNs) -Format im Allgemeine AWS-Referenz. Weitere Informationen zu Regionen finden Sie unter Über AWS Regionen auch in der Allgemeine AWS-Referenz.

Anzeigen von Informationen zum letzten Zugriff für IAM

Sie können die zuletzt aufgerufenen Informationen für IAM mithilfe der AWS API AWS Management Console AWS CLI, oder anzeigen. Hier finden Sie die Liste der Services und deren Aktionen, für die Informationen zum letzten Zugriff angezeigt werden. Weitere Informationen zu den Informationen zum letzten Zugriff finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Sie können Informationen für die folgenden Ressourcentypen in IAM anzeigen. In jedem Fall enthalten die Informationen die zulässigen Services für den jeweiligen Berichtszeitraum:

- Benutzer – Zeigt den letzten Zugriffsversuch des Benutzers auf jeden zulässigen Service an.
- Benutzergruppe – Zeigt Informationen zum letzten Zugriffsversuch auf den Service durch ein Mitglied der Gruppe an. Dieser Bericht enthält auch die Gesamtzahl der Mitglieder, die versucht haben, auf den Service zuzugreifen.
- Rolle – Zeigt den Zeitpunkt des letzten Zugriffsversuchs auf jeden zulässigen Service durch einen Benutzer mithilfe dieser Rolle an.
- Richtlinie – Zeigt Informationen zum letzten Zugriffsversuch auf jeden zulässigen Service durch einen Benutzer oder eine Rolle an. Dieser Bericht enthält auch die Gesamtzahl der Entitäten, die versucht haben, auf den Service zuzugreifen.

> **Note**
>
> Bevor Sie die Zugriffsinformationen für eine Ressource in IAM anzeigen, stellen Sie sicher, dass Sie den Berichtszeitraum, die gemeldeten Entitäten und die ausgewerteten Richtlinientypen für Ihre Informationen verstehen. Weitere Details finden Sie unter the section called “Wissenswertes über die Informationen zum letzten Zugriff”.

Anzeigen von Informationen für IAM (Konsole)

Sie können Informationen zum letzten Zugriff für IAM auf der Registerkarte Access Advisor (Advisor aufrufen) in der IAM-Konsole anzeigen.

So zeigen Sie Informationen für IAM an (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. Wählen Sie im Navigationsbereich entweder Groups (Gruppen), Users (Benutzer), Roles (Rollen) oder Policies (Richtlinien).
3. Wählen Sie einen Benutzer, eine Gruppe, eine Rolle oder einen Richtliniennamen aus, um die Seite Summary (Details) zu öffnen, und klicken Sie auf die Registerkarte Access Advisor. Zeigen Sie die folgenden Informationen basierend auf der gewählten Ressource an:

   - Group (Gruppe) – Zeigt die Liste der Services an, auf die Gruppenmitglieder (Benutzer) zugreifen können. Sie können auch anzeigen, wann ein Mitglied zuletzt auf den Service zugegriffen hat, welche Gruppenrichtlinien es verwendet hat, und welches Gruppenmitglied die Anforderung gestellt hat. Wählen Sie den Namen der Richtlinie aus, um zu erfahren, ob es sich um eine verwaltete Richtlinie oder eine eingebundene Gruppenrichtlinie handelt. Wählen Sie den Namen des Gruppenmitglieds aus, um alle Mitglieder der Gruppe und deren letztes Service-Zugriffsdatum anzuzeigen.
   - User (Benutzer) – Zeigt die Liste der Services an, auf die der Benutzer zugreifen kann. Sie können auch anzeigen, wann der Benutzer zuletzt auf den Service zugegriffen hat und welche Richtlinien dem Benutzer aktuell zugeordnet sind. Wählen Sie den Namen der Richtlinie aus, um zu erfahren, ob es sich um eine verwaltete Richtlinie, eine Inline-Benutzerrichtlinie oder eine Inline-Richtlinie für die Gruppe handelt.
   - Rolle – Zeigen Sie die Liste der Services, auf die die Rolle zugreifen kann, das letzte Service-Zugriffsdatum der Rolle und die verwendeten Richtlinien an. Wählen Sie den

Namen der Richtlinie aus, um zu erfahren, ob es sich um eine verwaltete Richtlinie oder eine eingebundene Rollenrichtlinie handelt.

- Policy (Richtlinie) – Zeigt die Liste der Services mit zulässigen Aktionen in der Richtlinie an. Sie können auch anzeigen, wann die Richtlinie zuletzt für den Zugriff auf den Service verwendet wurde, und welche Entität (Benutzer oder Rolle) die Richtlinie verwendet hat. Das Datum des letzten Zugriffs umfasst auch den Zeitpunkt, zu dem der Zugriff auf diese Richtlinie über eine andere Richtlinie gewährt wurde. Wählen Sie den Namen der Entität aus, um zu erfahren, welchen Entitäten diese Richtlinie angefügt ist und wann diese zuletzt auf den Service zugegriffen haben.

4. Wählen Sie in der Spalte Service der Tabelle den Namen eines der Services, der Informationen über die zuletzt aufgerufene Aktion enthält, um eine Liste der Verwaltungsaktionen anzuzeigen, auf die IAM-Entitäten versucht haben zuzugreifen. Sie können die AWS-Region und einen Zeitstempel anzeigen, der anzeigt, wann jemand zuletzt versucht hat, die Aktion auszuführen.
5. Die Spalte Letzter Zugriff wird für Services und Verwaltungsaktionen der Services angezeigt, die Informationen zur Aktion enthalten, auf die zuletzt zugegriffen wurde. Überprüfen Sie die folgenden möglichen Ergebnisse, die in dieser Spalte zurückgegeben werden. Diese Ergebnisse hängen davon ab, ob ein Dienst oder eine Aktion zugelassen ist, ob auf sie zugegriffen wurde und ob die Informationen, auf die zuletzt zugegriffen wurde AWS , nachverfolgt werden.

   vor <number of> Tagen

   Die Anzahl der Tage, seitdem der Service oder die Aktion im Nachverfolgungszeitraum verwendet wurde. Der Nachverfolgungszeitraum für Services erstreckt sich über die letzten 400 Tage. Der Nachverfolgungszeitraum für Amazon S3 Aktionen begann am 12. April 2020. Der Nachverfolgungszeitraum für die Verwaltungsmaßnahmen von Amazon EC2, IAM und Lambda begann am 7. April 2021. Der Erfassungszeitraum für alle anderen Services begann am 23. Mai 2023. Weitere Informationen zu den jeweiligen AWS-Region Startdaten der Nachverfolgung finden Sie unterWo werden die AWS zuletzt aufgerufenen Informationen aufgezeichnet.

   Kein Zugriff im Nachverfolgungszeitraum

   Der nachverfolgte Service oder die Aktion wurde im Nachverfolgungszeitraum nicht von einer Entität verwendet.

   Es ist möglich, dass Sie Berechtigungen für eine Aktion haben, die nicht in der Liste angezeigt wird. Dies kann vorkommen, wenn die Nachverfolgungsinformationen für die Aktion derzeit

nicht in AWS enthalten sind. Sie sollten keine Berechtigungsentscheidungen nur auf der Grundlage des Fehlens von Nachverfolgungsinformationen treffen. Stattdessen empfehlen wir, diese Informationen zu verwenden, um Ihre allgemeine Strategie für die Gewährung der geringsten Privilegien zu unterstützen. Überprüfen Sie Ihre Richtlinien, um zu bestätigen, dass die Zugriffsebene angemessen ist.

Anzeigen von Informationen für IAM (AWS CLI)

Sie können den verwenden AWS CLI , um Informationen darüber abzurufen, wann eine IAM-Ressource zum letzten Mal verwendet wurde, um auf AWS Services und Amazon S3-, Amazon EC2-, IAM- und Lambda-Aktionen zuzugreifen. Bei einer IAM-Ressource kann es sich um einen Benutzer, eine Gruppe, eine Rolle oder eine Richtlinie handeln.

So zeigen Sie Informationen für IAM (AWS CLI)

1. Erstellen Sie einen Bericht. Die Anforderung muss den ARN der IAM-Ressource (Benutzer, Gruppe, Rolle oder Richtlinie) enthalten, für die Sie den Bericht erstellen möchten. Sie können die Granularität angeben, die Sie im Bericht generieren möchten, um Zugriffsdetails nur für Services oder für Services und Aktionen anzuzeigen. Die Anforderung gibt eine `job-id` zurück, die Sie in den Operationen `get-service-last-accessed-details` und `get-service-last-accessed-details-with-entities` nutzen können, um den `job-status` bis zum Abschluss des Auftrags zu überwachen.

   - aws-IAM-Einzelheiten generate-service-last-accessed

2. Rufen Sie Details zum Bericht mithilfe des Parameters `job-id` aus dem vorherigen Schritt ab.

   - aws iam -details get-service-last-accessed

   Mit dieser Operation werden abhängig von der Art der Ressource und der Granularität, die Sie in der Operation `generate-service-last-accessed-details` angefordert haben, die folgenden Informationen zurückgegeben:

   - Benutzer – Gibt eine Liste der Services zurück, auf die der angegebene Benutzer zugreifen kann. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Zugriffsversuchs des Benutzers sowie den ARN des Benutzers zurück.
   - Benutzergruppe – Gibt eine Liste der Services zurück, auf die Mitglieder der angegebenen Benutzergruppe mithilfe der an die Benutzergruppe angefügten Richtlinien zugreifen

können. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Zugriffsversuchs durch ein Mitglied der Gruppe (Benutzer) zurück. Außerdem werden der ARN dieses Benutzers sowie die Gesamtzahl der Gruppenmitglieder, die versucht haben, auf den Service zuzugreifen, zurückgegeben. Verwenden Sie den GetServiceLastAccessedDetailsWithEntitiesVorgang, um eine Liste aller Mitglieder abzurufen.

- Rolle – Gibt eine Liste der Services zurück, auf die die angegebene Rolle zugreifen kann. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Zugriffsversuchs der Rolle sowie den ARN der Rolle zurück.
- Richtlinie – Gibt eine Liste der Services zurück, auf die die angegebene Richtlinie Zugriff gewährt. Die Operation gibt für jeden Service Datum und Uhrzeit des letzten Zugriffsversuch auf den Service durch eine Entität (Benutzer oder Rolle) mithilfe der Richtlinie zurück. Außerdem werden der ARN der Entität sowie die Gesamtzahl der Entitäten, die versucht haben, auf den Service zuzugreifen, zurückgegeben.

3. Weitere Informationen zu den Entitäten, die mithilfe von Gruppen- oder Richtlinienberechtigungen versucht haben, auf einen bestimmten Service zuzugreifen. Diese Operation gibt eine Liste der Entitäten einschließlich ARN, ID, Name, Pfad, Typ (Benutzer oder Rolle) und letztem Service-Zugriffsdatum jeder Entität zurück. Sie können diese Operation auch für Benutzer und Rollen verwenden. Es werden jedoch nur Informationen zu dieser Entität zurückgegeben.

   - war mein Ziel - get-service-last-accessed details-with-entities

4. Weitere Informationen zu den identitätsbasierten Richtlinien, die eine Identität (Benutzer, Gruppe oder Rolle) verwendet hat, um auf einen bestimmten Service zuzugreifen. Wenn Sie eine Identität und einen Service angeben, gibt diese Operation eine Liste der Berechtigungsrichtlinien zurück, die die Identität nutzen kann, um auf den angegebenen Service zuzugreifen. Diese Operation gibt den aktuellen Status der Richtlinien zurück und ist unabhängig von dem erzeugten Bericht. Es werden auch keine anderen Richtlinientypen wie ressourcenbasierte Richtlinien, Zugriffskontrolllisten, AWS Organizations -Richtlinien,IAM-Berechtigungsgrenzen oder Sitzungsrichtlinien zurückgegeben. Weitere Informationen finden Sie unter Richtlinientypen oder Auswerten von Richtlinien in einem einzelnen Konto.

   - aws list-policies-granting-service iam -access

Informationen für IAM (API) anzeigen AWS

Sie können die AWS API verwenden, um Informationen darüber abzurufen, wann eine IAM-Ressource zum letzten Mal verwendet wurde, um auf AWS Services und Amazon S3-, Amazon EC2-, IAM- und Lambda-Aktionen zuzugreifen. Bei einer IAM-Ressource kann es sich um einen Benutzer, eine Gruppe, eine Rolle oder eine Richtlinie handeln. Sie können die Granularität angeben, die Sie im Bericht generieren möchten, um Details nur für Services oder für Services und Aktionen anzuzeigen.

Um Informationen für IAM (API) anzuzeigen AWS

1. Erstellen Sie einen Bericht. Die Anforderung muss den ARN der IAM-Ressource (Benutzer, Gruppe, Rolle oder Richtlinie) enthalten, für die Sie den Bericht erstellen möchten. Es wird eine `JobId` zurückgegeben, die Sie in den Operationen `GetServiceLastAccessedDetails` und `GetServiceLastAccessedDetailsWithEntities` nutzen können, um den `JobStatus` bis zum Abschluss des Auftrags zu überwachen.

   - GenerateServiceLastAccessedDetails
2. Rufen Sie Details zum Bericht mithilfe des Parameters `JobId` aus dem vorherigen Schritt ab.

   - GetServiceLastAccessedDetails

   Mit dieser Operation werden abhängig von der Art der Ressource und der Granularität, die Sie in der Operation `GenerateServiceLastAccessedDetails` angefordert haben, die folgenden Informationen zurückgegeben:

   - Benutzer – Gibt eine Liste der Services zurück, auf die der angegebene Benutzer zugreifen kann. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Zugriffsversuchs des Benutzers sowie den ARN des Benutzers zurück.
   - Benutzergruppe – Gibt eine Liste der Services zurück, auf die Mitglieder der angegebenen Benutzergruppe mithilfe der an die Benutzergruppe angefügten Richtlinien zugreifen können. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Zugriffsversuchs durch ein Mitglied der Gruppe (Benutzer) zurück. Außerdem werden der ARN dieses Benutzers sowie die Gesamtzahl der Gruppenmitglieder, die versucht haben, auf den Service zuzugreifen, zurückgegeben. Verwenden Sie den GetServiceLastAccessedDetailsWithEntitiesVorgang, um eine Liste aller Mitglieder abzurufen.
   - Rolle – Gibt eine Liste der Services zurück, auf die die angegebene Rolle zugreifen kann. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Zugriffsversuchs der Rolle sowie den ARN der Rolle zurück.

- Richtlinie – Gibt eine Liste der Services zurück, auf die die angegebene Richtlinie Zugriff gewährt. Die Operation gibt für jeden Service Datum und Uhrzeit des letzten Zugriffsversuch auf den Service durch eine Entität (Benutzer oder Rolle) mithilfe der Richtlinie zurück. Außerdem werden der ARN der Entität sowie die Gesamtzahl der Entitäten, die versucht haben, auf den Service zuzugreifen, zurückgegeben.

3. Weitere Informationen zu den Entitäten, die mithilfe von Gruppen- oder Richtlinienberechtigungen versucht haben, auf einen bestimmten Service zuzugreifen. Diese Operation gibt eine Liste der Entitäten einschließlich ARN, ID, Name, Pfad, Typ (Benutzer oder Rolle) und letztem Service-Zugriffsdatum jeder Entität zurück. Sie können diese Operation auch für Benutzer und Rollen verwenden. Es werden jedoch nur Informationen zu dieser Entität zurückgegeben.

   - GetServiceLastAccessedDetailsWithEntities

4. Weitere Informationen zu den identitätsbasierten Richtlinien, die eine Identität (Benutzer, Gruppe oder Rolle) verwendet hat, um auf einen bestimmten Service zuzugreifen. Wenn Sie eine Identität und einen Service angeben, gibt diese Operation eine Liste der Berechtigungsrichtlinien zurück, die die Identität nutzen kann, um auf den angegebenen Service zuzugreifen. Diese Operation gibt den aktuellen Status der Richtlinien zurück und ist unabhängig von dem erzeugten Bericht. Es werden auch keine anderen Richtlinientypen wie ressourcenbasierte Richtlinien, Zugriffskontrolllisten, AWS Organizations -Richtlinien,IAM-Berechtigungsgrenzen oder Sitzungsrichtlinien zurückgegeben. Weitere Informationen finden Sie unter Richtlinientypen oder Auswerten von Richtlinien in einem einzelnen Konto.

   - ListPoliciesGrantingServiceAccess

So zeigen Sie Informationen zum letzten Zugriff für Organizations an

Sie können Informationen zum zuletzt aufgerufenen Dienst für die AWS Organizations Verwendung der IAM-Konsole oder AWS API anzeigen. AWS CLI Für wichtige Informationen über die Daten, die erforderlichen Berechtigungen, Fehlerbehebungen und unterstützte Regionen finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Wenn Sie sich mit den Anmeldeinformationen für das AWS Organizations Verwaltungskonto bei der IAM-Konsole anmelden, können Sie Informationen für jede Entität in Ihrer Organisation einsehen. Zu den Organizations einheiten gehören der Organisationsstamm, die Organisationseinheiten (OUs) und die Konten. Sie können die IAM-Konsole auch verwenden, um sich Informationen für

alle Service-Kontrollrichtlinien (Service Control Policies, SCPs) in Ihrer Organisation anzeigen zu lassen. IAM zeigt eine Liste der Services, die von allen SCPs, die auf die Entity angewendet werden, zugelassen sind. Für jeden Dienst können Sie die neuesten Kontoinformationen für die gewählte Organisationseinheit oder die Kinder der Einheit einsehen.

Wenn Sie die AWS API AWS CLI oder mit den Anmeldeinformationen für das Verwaltungskonto verwenden, können Sie einen Bericht für alle Entitäten oder Richtlinien in Ihrer Organisation erstellen. Ein programmgesteuerter Bericht für eine Entity enthält eine Liste der Dienste, die von allen SCPs, die auf die Entity angewendet werden, zugelassen sind. Für jeden Service enthält der Bericht die aktuellen Aktivitäten für Konten in der angegebenen Organizations-Entity oder der Unterstruktur der Entität.

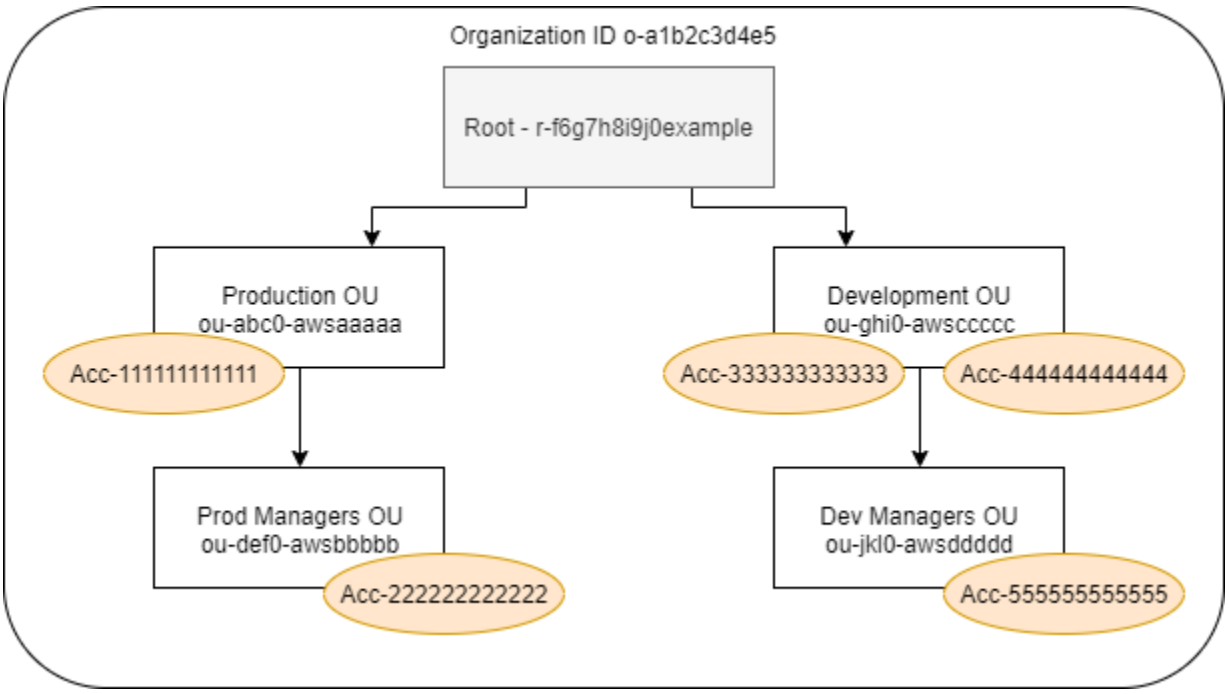
Wenn Sie einen programmatischen Bericht für eine Richtlinie erstellen, müssen Sie eine Organisationseinheit angeben. Dieser Bericht enthält eine Liste der Services, die im Rahmen der angegebenen SCP zugelassen sind. Für die einzelnen Services sind die neuesten Kontoaktivitäten der Entity oder der untergeordneten Elemente der Entity enthalten, denen aufgrund der Richtlinie eine Berechtigung erteilt wird. Weitere Informationen finden Sie unter aws iam generate-organizations-access-report oder GenerateOrganizationsAccessReport.

Bevor Sie sich den Bericht anzeigen lassen, stellen Sie sicher, dass Sie die Anforderungen an das Hauptkonto und die Daten, den Berichtszeitraum, die Entitäten, über die berichtet wird, als auch die ausgewerteten Richtlinientypen verstehen. Weitere Details finden Sie unter the section called “Wissenswertes über die Informationen zum letzten Zugriff”.

Der AWS Organizations -Entitätspfad

Wenn Sie die AWS API AWS CLI oder verwenden, um einen AWS Organizations Zugriffsbericht zu generieren, müssen Sie einen Entitätspfad angeben. Ein Pfad ist eine Textdarstellung der Struktur einer Organizations-Entität.

Sie können einen Entitätspfad mithilfe der bekannten Struktur Ihrer Organisation erstellen. Gehen Sie beispielsweise davon aus, dass Sie über die folgende Organisationsstruktur verfügen AWS Organizations.



Der Pfad für die Organisationseinheit des Dev Managers (Entwicklungsmanagers) wird mithilfe der IDs der Organisation, des Stammes und aller Organisationseinheiten im Pfad bis einschließlich der Organisationseinheit erstellt.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsddddd/
```

Der Pfad für das Konto in der Organisationseinheit Production (Produktion) wird mithilfe der IDs der Organisation, des Stammes, der Organisationseinheit und der Kontonummer erstellt.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> Organisations-IDs sind global eindeutig, Organisationeinheiten-IDs und Stamm-IDs sind jedoch nur innerhalb einer Organisation eindeutig. Dies bedeutet, dass keine zwei Organisationen dieselbe Organisations-ID verwenden. Eine andere Organisation verfügt jedoch möglicherweise über eine Organisationseinheit oder ein Stammverzeichnis mit derselben ID wie Ihre. Es wird empfohlen, immer die Organisations-ID anzugeben, wenn Sie eine Organisationseinheit oder ein Stammverzeichnis angeben.

Anzeigen von Informationen für Organizations (Konsole)

Sie können die IAM-Konsole verwenden, um Informationen zum letzten Servicezugriff für den Stamm, die Organisationseinheit, das Konto oder die Richtlinie anzeigen zu lassen.

So zeigen Sie Informationen für das Stammverzeichnis (Konsole) an:

1. Melden Sie sich AWS Management Console mit den Anmeldeinformationen für das Verwaltungskonto using Organizations an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich unter dem Abschnitt Access reports (Zugriffsberichte) die Option Organization activity (Organisationsaktivität).
3. Wählen Sie auf der Seite Organization activity (Organisationsaktivität) die Option Root (Stamm).
4. Überprüfen Sie auf der Registerkarte Details and activity (Details und Aktivitäten) den Bereich Service access report (Service-Zugriffsbericht). Die Informationen enthalten eine Liste der Services, die aufgrund der Richtlinien, die dem Stamm direkt angefügt wurden, zulässig sind. Die Informationen zeigen Ihnen, von welchem Konto aus und wann zuletzt auf den Service zugegriffen wurde. Um weitere Informationen darüber zu erhalten, welcher Auftraggeber auf den Service zugegriffen hat, melden Sie sich als Administrator bei diesem Konto an, und zeigen Sie die Informationen zum letzten IAM-Servicezugriff an.
5. Wählen Sie die Registerkarte Attached SCPs (Angehängte SCPs), um die Liste der Dienstkontrollrichtlinien (SCPs) anzuzeigen, die an den Stamm angehängt sind. IAM zeigt Ihnen die Anzahl der Zielentitäten an, denen die einzelnen Richtlinien zugeordnet sind. Sie können diese Informationen verwenden, um zu entscheiden, welche SCP Sie überprüfen möchten.
6. Wählen Sie den Namen einer SCP auf, um sich alle Services anzeigen zu lassen, die von der Richtlinie zugelassen sind. Lassen Sie sich für jeden Service anzeigen, von welchem Konto und wann auf den Service zuletzt zugegriffen wurde.
7. Klicken Sie auf Bearbeiten in AWS Organizations, um weitere Details anzuzeigen und den SCP in der Organizationsationskonsole zu bearbeiten. Weitere Informationen finden Sie unter -Over-the-Air-Updates im AWS Organizations -Leitfaden.

So zeigen Sie Informationen zu einer Organisationseinheit oder einem Konto an (Konsole):

1. Melden Sie sich AWS Management Console mit den Anmeldeinformationen für das Verwaltungskonto using Organizations an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

2. Wählen Sie im Navigationsbereich unter dem Abschnitt Access reports (Zugriffsberichte) die Option Organization activity (Organisationsaktivität).
3. Erweitern Sie auf der Seite Organization activity (Organizationsaktivität) die Struktur Ihrer Organisation. Wählen Sie anschließend den Namen der Organisationseinheit oder jedes beliebigen Kontos aus, welches Sie anzeigen möchten, mit Ausnahme des Hauptkontos.
4. Überprüfen Sie auf der Registerkarte Details and activity (Details und Aktivitäten) den Bereich Service access report (Service-Zugriffsbericht). Die Informationen enthalten eine Liste der Services, die im Rahmen der SCPs, die der Organisationseinheit oder dem Konto und allen übergeordneten Elementen angefügt wurden, zulässig sind. Die Informationen zeigen Ihnen, von welchem Konto aus und wann zuletzt auf den Service zugegriffen wurde. Um weitere Informationen darüber zu erhalten, welcher Auftraggeber auf den Service zugegriffen hat, melden Sie sich als Administrator bei diesem Konto an, und zeigen Sie die Informationen zum letzten IAM-Servicezugriff an.
5. Wählen Sie die Registerkarte Angehängte SCPs, um die Liste der Dienstkontrollrichtlinien (SCPs) anzuzeigen, die direkt mit der Organisationseinheit oder dem Konto verbunden sind. IAM zeigt Ihnen die Anzahl der Zielentitäten an, denen die einzelnen Richtlinien zugeordnet sind. Sie können diese Informationen verwenden, um zu entscheiden, welche SCP Sie überprüfen möchten.
6. Wählen Sie den Namen einer SCP auf, um sich alle Services anzeigen zu lassen, die von der Richtlinie zugelassen sind. Lassen Sie sich für jeden Service anzeigen, von welchem Konto und wann auf den Service zuletzt zugegriffen wurde.
7. Klicken Sie auf Bearbeiten in AWS Organizations, um weitere Details anzuzeigen und den SCP in der Organizationsationskonsole zu bearbeiten. Weitere Informationen finden Sie unter -Over-the-Air-Updates im AWS Organizations -Leitfaden.

So zeigen Sie Informationen zum Hauptkonto an (Konsole):

1. Melden Sie sich AWS Management Console mit den Anmeldeinformationen für das Verwaltungskonto using Organizations an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich unter dem Abschnitt Access reports (Zugriffsberichte) die Option Organization activity (Organisationsaktivität).
3. Erweitern Sie auf der Seite Organization activity (Organisationsaktivität) die Struktur Ihrer Organisation und wählen Sie den Namen Ihres Hauptkontos.

4. Überprüfen Sie auf der Registerkarte Details and activity (Details und Aktivitäten) den Bereich Service access report (Service-Zugriffsbericht). Die Informationen enthalten eine Liste aller AWS -Services. Das Hauptkonto ist nicht durch SCPs eingeschränkt. Die Informationen geben darüber Auskunft, ob und wann das Konto zuletzt auf den Service zugegriffen hat. Um weitere Informationen darüber zu erhalten, welcher Auftraggeber auf den Service zugegriffen hat, melden Sie sich als Administrator bei diesem Konto an, und zeigen Sie die Informationen zum letzten IAM-Servicezugriff an.
5. Wählen Sie die Registerkarte Attached SCPs (Angefügte SCPs) aus, um zu bestätigen, dass keine angefügten SCPs vorhanden sind, da es sich bei dem Konto um das Hauptkonto handelt.

So zeigen Sie Informationen zu einer Richtlinie an (Konsole):

1. Melden Sie sich AWS Management Console mit den Anmeldeinformationen für das Verwaltungskonto using Organizations an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich unter dem Abschnitt Access reports (Zugriffsberichte) die Option Service control policies (SCPs) (Service-Kontrollrichtlinien (SCPs)).
3. Auf der Seite Service Control Policies (SCPs) (Service-Kontrollrichtlinien) finden Sie eine Liste der Richtlinien Ihrer Organisation. Sie können die Anzahl der Ziel-Entitäten anzeigen lassen, an die die einzelnen Richtlinien angefügt sind.
4. Wählen Sie den Namen einer SCP auf, um sich alle Services anzeigen zu lassen, die von der Richtlinie zugelassen sind. Lassen Sie sich für jeden Service anzeigen, von welchem Konto und wann auf den Service zuletzt zugegriffen wurde.
5. Klicken Sie auf Bearbeiten in AWS Organizations, um weitere Details anzuzeigen und den SCP in der Organizationsationskonsole zu bearbeiten. Weitere Informationen finden Sie unter -Over-the-Air-Updates im AWS Organizations -Leitfaden.

Anzeigen von Informationen für Organizations (AWS CLI)

Sie können den verwenden AWS CLI , um Informationen zum letzten Zugriff auf den Dienst für den Stamm, die Organisationseinheit, das Konto oder die Richtlinie Ihres Unternehmens abzurufen.

Um Informationen über den letzten Zugriff auf den Organisationsdienst anzuzeigen (AWS CLI)

1. Verwenden Sie die Anmeldedaten für Ihr Organisationsmanagementkonto mit den erforderlichen IAM- und Organisationsberechtigungen und stellen Sie sicher, dass SCPs für Ihr

Stammverzeichnis aktiviert sind. Weitere Informationen finden Sie unter Wissenswertes über die Informationen zum letzten Zugriff.

2. Erstellen Sie einen Bericht. Die Anforderung muss den Pfad der Organizations-Entität umfassen (Stamm, Organisationseinheit oder Konto), für die Sie einen Bericht erstellen möchten. Sie können optional einen `organization-policy-id`-Parameter einschließen, um den Bericht für eine bestimmte Richtlinie zu sehen. Der Befehl gibt eine `job-id` zurück, die Sie im Anschluss daran mit dem `get-organizations-access-report`-Befehl verwenden können, um den `job-status` zu überwachen, bis der Auftrag abgeschlossen ist.

   - war ich generate-organizations-access-report

3. Rufen Sie Details zum Bericht mithilfe des Parameters `job-id` aus dem vorherigen Schritt ab.

   - war ich get-organizations-access-report

   Dieser Befehl gibt eine Liste der Dienste zurück, auf die Entity-Mitglieder zugreifen können. Für jeden Service gibt der Befehl das Datum und die Uhrzeit des letzten Versuchs eines Kontomitglieds sowie den Entity-Pfad des Kontos zurück. Außerdem wird die Gesamtanzahl der Dienste angezeigt, auf die zugegriffen werden kann, als auch die Anzahl der Services, auf die nicht zugegriffen wurde. Wenn Sie den optionalen `organizations-policy-id`-Parameter angegeben haben, handelt es sich bei den Diensten, die für den Zugriff verfügbar sind, um die Dienste, die von der angegebenen Richtlinie zugelassen sind.

Informationen für Organizations anzeigen (AWS API)

Sie können die AWS API verwenden, um Informationen zum zuletzt aufgerufenen Dienst für den Stamm, die Organisationseinheit, das Konto oder die Richtlinie Ihres Unternehmens abzurufen.

So zeigen Sie Informationen zum zuletzt aufgerufenen Service der Organizations an (AWS API)

1. Verwenden Sie die Anmeldedaten für Ihr Organisationsmanagementkonto mit den erforderlichen IAM- und Organisationsberechtigungen und stellen Sie sicher, dass SCPs für Ihr Stammverzeichnis aktiviert sind. Weitere Informationen finden Sie unter Wissenswertes über die Informationen zum letzten Zugriff.
2. Erstellen Sie einen Bericht. Die Anforderung muss den Pfad der Organizations-Entität umfassen (Stamm, Organisationseinheit oder Konto), für die Sie einen Bericht erstellen möchten. Sie können optional einen `OrganizationsPolicyId`-Parameter einschließen, um den Bericht für eine bestimmte Richtlinie zu sehen. Durch die Operation wird eine `JobId` zurückgegeben,

die Sie in der Operation `GetOrganizationsAccessReport` verwenden können, um den `JobStatus` zu überwachen, bis der Auftrag abgeschlossen ist.

- GenerateOrganizationsAccessReport

3. Rufen Sie Details zum Bericht mithilfe des Parameters `JobId` aus dem vorherigen Schritt ab.

   - GetOrganizationsAccessReport

   Diese Operation gibt eine Liste der Dienste zurück, auf die von den Entity-Mitgliedern zugegriffen werden kann. Für jeden Service gibt die Operation das Datum und die Uhrzeit des letzten Versuchs eines Kontomitglieds sowie den Entity-Pfad des Kontos zurück. Außerdem wird die Gesamtanzahl der Dienste angezeigt, auf die zugegriffen werden kann, als auch die Anzahl der Services, auf die nicht zugegriffen wurde. Wenn Sie den optionalen `OrganizationsPolicyId`-Parameter angegeben haben, handelt es sich bei den Diensten, die für den Zugriff verfügbar sind, um die Dienste, die von der angegebenen Richtlinie zugelassen sind.

Beispielszenarien für die Verwendung von Informationen zum letzten Zugriff

Sie können die Informationen, auf die zuletzt zugegriffen wurde, verwenden, um Entscheidungen über die Berechtigungen zu treffen, die Sie Ihren IAM-Entitäten oder AWS Organizations -Entitäten gewähren. Weitere Informationen finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

> **Note**
>
> Bevor Sie die Zugriffsinformationen für eine Entität oder Richtlinie in IAM einsehen, sollten Sie sicherstellen AWS Organizations, dass Sie den Berichtszeitraum, die gemeldeten Entitäten und die bewerteten Richtlinientypen für Ihre Daten kennen. Weitere Details finden Sie unter the section called “Wissenswertes über die Informationen zum letzten Zugriff”.

Sie als Administrator sind dafür zuständig, die richtige Ausgewogenheit zwischen Zugriffsfähigkeit und geringsten Berechtigungen zu finden, wie sie für Ihre Organisation geeignet ist.

Verwenden von Informationen zum Reduzieren von Berechtigungen für eine IAM-Gruppe

Sie können die Daten des letzten Servicezugriffs verwenden, um IAM-Gruppenberechtigungen so zu reduzieren, dass nur die Services enthalten sind, die Ihre Benutzer wirklich benötigen. Diese Methode ist ein wichtiger Schritt beim Erteilen von geringsten Rechten auf Service-Ebene.

Paulo Santos ist beispielsweise der Administrator, der für die Definition von AWS Benutzerberechtigungen für Example Corp. zuständig ist. Dieses Unternehmen hat gerade damit begonnen, die Software zu nutzen AWS, und das Softwareentwicklungsteam hat noch nicht definiert, welche AWS Dienste es nutzen wird. Paulo möchte dem Team Berechtigung für den Zugriff nur auf die Services geben, die benötigt werden, aber da dies noch nicht definiert ist, gibt er dem Team vorübergehend PowerUser-Berechtigungen. Anschließend verwendet er die Informationen zum letzten Zugriff, um die Berechtigungen der Gruppe zu reduzieren.

Paulo erstellt mithilfe des folgenden JSON-Texts eine verwaltete Richtlinie mit dem Namen `ExampleDevelopment`. Er weist sie dann einer Gruppe mit dem Namen `Development` zu und fügt der Gruppe alle Entwickler hinzu.

> **Note**
>
> Paulos Hauptbenutzer benötigen möglicherweise `iam:CreateServiceLinkedRole`-Berechtigungen, um einige Dienste und Funktionen nutzen zu können. Er weiß, dass das Hinzufügen dieser Berechtigung es den Benutzern ermöglicht, eine serviceverknüpfte Rolle zu erstellen. Er akzeptiert dieses Risiko für seine Hauptbenutzer.

```
{

    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo entscheidet sich, 90 Tage warten, bevor er mithilfe der AWS Management Console die Informationen zum letzten Zugriffs für die `Development`-Gruppe anzeigt. Er zeigt die Liste der Services an, auf die Mitglieder der Gruppe zugegriffen haben. Er erfährt, dass die Benutzer in der letzten Woche auf fünf Dienste zugegriffen haben: AWS CloudTrail Amazon CloudWatch Logs, Amazon EC2 und Amazon S3. AWS KMS Sie haben bei der ersten Evaluierung auf einige andere Dienste zugegriffen AWS, aber seitdem nicht mehr.

Paulo beschließt, die Richtlinienberechtigungen zu reduzieren, so dass nur diese fünf Services und die erforderlichen IAM- und Organizations-Aktionen enthalten sind. Er bearbeitet die `ExampleDevelopment`-Richtlinie mithilfe des folgenden JSON-Texts.

> **Note**
>
> Paulos Hauptbenutzer benötigen möglicherweise `iam:CreateServiceLinkedRole`-Berechtigungen, um einige Dienste und Funktionen nutzen zu können. Er weiß, dass das Hinzufügen dieser Berechtigung es den Benutzern ermöglicht, eine serviceverknüpfte Rolle zu erstellen. Er akzeptiert dieses Risiko für seine Hauptbenutzer.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
```

```
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Um die Anzahl der Zugriffsberechtigungen weiter zu reduzieren, kann Paulo die Ereignisse des Accounts im AWS CloudTrail Event-Verlauf einsehen. Dort kann er detaillierte Ereignisinformationen anzeigen, die er verwenden kann, um Berechtigungen der Richtlinie so zu reduzieren, dass nur die Aktionen und Ressourcen enthalten sind, die die Entwickler benötigen. Weitere Informationen finden Sie im AWS CloudTrail Benutzerhandbuch unter CloudTrail Ereignisse in der CloudTrail Konsole anzeigen.

Verwenden von Informationen zum Reduzieren von Berechtigungen für einen IAM-Benutzer

Sie können die Informationen des letzten Servicezugriffs verwenden, um die Berechtigungen für einen einzelnen IAM-Benutzer zu reduzieren.

Martha Rivera ist beispielsweise als IT-Administratorin dafür verantwortlich, dass die Mitarbeiter in ihrem Unternehmen nicht über zu viele Berechtigungen verfügen. AWS Im Rahmen einer regelmäßigen Sicherheitsprüfung überprüft sie die Berechtigungen aller IAM-Benutzer. Einer dieser Benutzer ist ein Anwendungsentwickler namens Nikhil Jayashankar, der vorher die Funktion eines Sicherheitsingenieurs innehatte. Aufgrund der Änderung der Stellenanforderungen ist Nikhil Mitglied sowohl der `app-dev`-Gruppe als auch der `security-team`-Gruppe. Die `app-dev`-Gruppe für seinen neuen Auftrag gewährt Berechtigungen für mehrere Dienste, darunter Amazon EC2, Amazon EBS, Auto Scaling, Amazon S3, Route 53 und Elastic Transcoder. Die `security-team` Gruppe für seinen alten Job erteilt IAM und Berechtigungen. CloudTrail

Martha meldet sich als Administrator bei der IAM-Konsole an und wählt Users (Benutzer), anschließend den Namen `nikhilj` und dann die Registerkarte Access Advisor aus.

Martha überprüft die Spalte Zuletzt aufgerufen und stellt fest, dass Nikhil in letzter Zeit nicht auf IAM, Route 53 CloudTrail, Amazon Elastic Transcoder und eine Reihe anderer Dienste zugegriffen hat. AWS Nikhil hat Zugriff auf Amazon S3. Martha wählt S3 aus der Liste der Services und erfährt, dass Nikhil in den letzten zwei Wochen einige S3-`List`-Aktionen durchgeführt hat. In ihrem Unternehmen bestätigt Martha, dass Nikhil keine geschäftliche Notwendigkeit hat, auf IAM zuzugreifen, und zwar auch deshalb, weil er kein Mitglied des CloudTrail internen Sicherheitsteams mehr ist.

Martha ist jetzt bereit, zu dem Service und den Aktionen, auf die zuletzt zugegriffen wurde, zu handeln. Im Gegensatz zu der Gruppe im vorherigen Beispiel können auf einen IAM-Benutzer wie `nikhilj` mehrere Richtlinien zutreffen und er kann Mitglied mehrerer Gruppen sein. Martha muss vorsichtig vorgehen, damit sie nicht versehentlich den Zugriff für `nikhilj` oder andere Gruppenmitglieder unterbricht. Sie muss nicht nur herausfinden, welchen Zugriff Nikhil haben sollte, sie muss auch bestimmen, wie er diese Berechtigungen erhält.

Martha wählt die Registerkarte Permissions (Berechtigungen) aus, wo sie anzeigt, welche Richtlinien `nikhilj` direkt zugewiesen sind und welche aus einer Gruppe angefügt sind. Sie erweitert die einzelnen Richtlinien und zeigt die Richtlinienübersicht an, um zu erfahren, welche Richtlinie Zugriff auf die Services gewährt, die Nikhil nicht verwendet:

- IAM — Die `IAMFullAccess` AWS verwaltete Richtlinie ist direkt an die Gruppe angehängt `nikhilj` und an diese angehängt. `security-team`
- CloudTrail — Die `AWS CloudTrailReadOnlyAccess` AWS verwaltete Richtlinie ist der `security-team` Gruppe zugeordnet.
- Route 53 – Die vom Kunden verwaltete `App-Dev-Route53`-Richtlinie ist der `app-dev`-Gruppe angefügt.
- Elastic Transcoder – Die vom Kunden verwaltete `App-Dev-ElasticTranscoder`-Richtlinie ist der `app-dev`-Gruppe angefügt.

Martha beschließt, die `IAMFullAccess` AWS verwaltete Richtlinie zu entfernen, die direkt angehängt ist. `nikhilj` Außerdem entfernt sie Nikhils Mitgliedschaft bei der `security-team`-Gruppe. Durch diese beiden Aktionen wird der unnötige Zugriff auf IAM und entfernt. CloudTrail

Nikhils Berechtigungen für den Zugriff auf Route 53 und Elastic Transcoder werden von der `app-dev`-Gruppe gewährt. Obwohl Nikhil diese Services nicht verwendet, kann es sein, dass andere Mitglieder der Gruppe das tun. Martha überprüft die Informationen über den letzten Zugriff auf

die `app-dev`-Gruppe und erfährt, dass mehrere Mitglieder kürzlich auf Route 53 und Amazon S3 zugegriffen haben. Aber keine Gruppenmitglieder haben im letzten Jahr auf Elastic Transcoder zugegriffen. Sie entfernt die kundenverwaltete Richtlinie `App-Dev-ElasticTranscoder` aus der Gruppe.

Martha überprüft dann die Informationen zum letzten Servicezugriff für die kundenverwaltete Richtlinie `App-Dev-ElasticTranscoder`. Sie stellt fest, dass die Richtlinie nicht an andere IAM-Identitäten angefügt ist. Sie stellt innerhalb ihres Unternehmens Nachforschungen an, um sicherzustellen, dass die Richtlinie auch in der Zukunft benötigt wird, und löscht sie dann.

Verwenden von Informationen vor dem Löschen von IAM-Ressourcen

Sie können die Informationen zum letzten Servicezugriff verwenden, bevor Sie eine IAM-Ressource löschen, um sicherzustellen, dass eine gewisse Zeit vergangen ist, seitdem jemand die Ressource zum letzten Mal verwendet hat. Dies gilt für Benutzer, Gruppen, Rollen und Richtlinien. Weitere Informationen zu diesen Aktionen finden Sie in den folgenden Themen:

- Benutzer – Löschen eines Benutzers
- Gruppen – Löschen einer Gruppe
- Rollen – Löschen einer Rolle
- Richtlinien – Löschen einer verwalteten Richtlinie (wodurch die Richtlinie auch von Identitäten getrennt wird)

Verwenden von Informationen vor dem Bearbeiten von IAM-Richtlinien

Sie können die Informationen zum letzten Zugriff für eine IAM-Identität (Benutzer, Gruppe oder Rolle) oder für eine IAM-Richtlinie verwenden, bevor Sie eine Richtlinie bearbeiten, die sich auf diese Ressource auswirkt. Dies ist wichtig, da Sie nicht den Zugriff für jemanden entfernen möchten, der sie verwendet.

Arnav Desai ist beispielsweise Entwickler und AWS Administrator für Example Corp. Als sein Team mit der Nutzung begann, gewährten sie allen Entwicklern Power-User-Zugriff AWS, sodass sie vollen Zugriff auf alle Dienste außer IAM und Organizations hatten. Als ersten Schritt zur Gewährung der geringsten Rechte möchte Arnav die verwenden, um die verwalteten Richtlinien in seinem Konto AWS CLI zu überprüfen.

Dazu listet Arnav zuerst die kundenverwalteten Berechtigungsrichtlinien in seinem Konto auf, die einer Identität zugewiesen sind, indem er den folgenden Befehl ausführt:

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

Aus der Antwort erfasst er den ARN für die einzelnen Richtlinien. Arnav generiert dann mit dem folgenden Befehl einen Bericht über die Informationen zum letzten Zugriff für jede Richtlinie.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Aus dieser Antwort erfasst er die ID des generierten Berichts aus dem `JobId`-Feld. Arnav fragt dann den folgenden Befehl ab, bis das `JobStatus`-Feld den Wert `COMPLETED` oder `FAILED` zurückgibt. Wenn der Auftrag fehlschlägt, erfasst er den Fehler.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Wenn der Auftrag den Status `COMPLETED` aufweist, analysiert Arnav den Inhalt des `ServicesLastAccessed`-Arrays im JSON-Format.

```
 "ServicesLastAccessed": [
        {
            "TotalAuthenticatedEntities": 1,
            "LastAuthenticated": 2018-11-01T21:24:33.222Z,
            "ServiceNamespace": "dynamodb",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
            "ServiceName": "Amazon DynamoDB"
        },

        {
            "TotalAuthenticatedEntities": 0,
            "ServiceNamespace": "ec2",
            "ServiceName": "Amazon EC2"
        },

        {
            "TotalAuthenticatedEntities": 3,
            "LastAuthenticated": 2018-08-25T15:29:51.156Z,
            "ServiceNamespace": "s3",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
            "ServiceName": "Amazon S3"
        }
```

```
    ]
```

Anhand dieser Informationen stellt Arnav fest, dass die `ExamplePolicy1`-Richtlinie den Zugriff auf drei Services erlaubt, Amazon DynamoDB, Amazon S3 und Amazon EC2. Der `IAMExampleUser`-Benutzer namens `IAMExampleRole` hat am 1. November als letztes versucht, auf DynamoDB zuzugreifen, und jemand hat am 25. August die IAM-Rolle verwendet, um auf Amazon S3 zuzugreifen. Darüber hinaus gibt es zwei weitere Entitys, die im letzten Jahr versucht haben, auf Amazon S3 zuzugreifen. Allerdings hat im letzten Jahr niemand versucht, auf Amazon EC2 zugreifen.

Dies bedeutet, dass Arnav die Amazon EC2-Aktionen aus der Richtlinie entfernen kann.
Arnav möchte das aktuelle JSON-Dokument für die Richtlinie überprüfen. Zuerst muss er die Versionsnummer der Richtlinie mit dem folgenden Befehl bestimmen.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Aus der Antwort erfasst Arnav die aktuelle Standardversionsnummer aus dem `Versions`-Array. Er verwendet dann diese Versionsnummer (`v2`), um das JSON-Richtliniendokument mit dem folgenden Befehl anzufordern.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav speichert das JSON-Richtliniendokument, das im `Document`-Feld des `PolicyVersion`-Arrays zurückgegeben wurde. Innerhalb des Richtliniendokuments sucht Arnav nach Aktionen mit dem `ec2`-Namespace. Wenn keine Aktionen von anderen Namespaces in der Richtlinie verbleiben, trennt Arnav die Richtlinie von den betroffenen Identitäten (Benutzern, Gruppen und Rollen). Im Anschluss daran löscht Arnav die Richtlinie. In diesem Fall umfasst die Richtlinie die Services Amazon DynamoDB und Amazon S3. Arnav entfernt daher die Amazon EC2-Aktionen aus dem Dokument und speichert die Änderungen. Er verwendet dann den folgenden Befehl, um die Richtlinie mit der neuen Version des Dokuments zu aktualisieren und diese Version als Standardrichtlinienversion festzulegen.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

Die `ExamplePolicy1`-Richtlinie ist jetzt aktualisiert und entfernt den Zugriff auf den unnötigen Amazon EC2-Service.

Andere IAM-Szenarien

Informationen darüber, wann eine IAM-Ressource (Benutzer, Gruppe, Rolle oder Richtlinie) zuletzt versucht hat, auf einen Service zuzugreifen, können Ihnen bei der Durchführung der folgenden Aufgaben helfen:

- Richtlinien – Bearbeiten einer bestehenden kundenverwalteten oder eingebundenen Richtlinie zum Entfernen von Berechtigungen
- Richtlinien – Konvertieren einer eingebundenen Richtlinie in eine verwaltete Richtlinie und anschließendes Löschen der Richtlinie
- Richtlinien – Hinzufügen einer ausdrücklichen Ablehnung zu einer bestehenden Richtlinie
- Richtlinien – Trennen einer verwalteten Richtlinie von einer Identität (Gruppe, Rolle oder Benutzer)
- Entitys – Festlegen einer Berechtigungsgrenze zum Steuern der maximalen Berechtigungen, die eine Entity (Benutzer oder Rolle) haben kann
- Gruppen – Entfernen von Benutzern aus einer Gruppe

Verwenden von Informationen zum Optimieren von Berechtigungen für eine Organisationseinheit

Sie können die Informationen zum letzten Zugriff verwenden, um die Berechtigungen für eine Organisationseinheit (OU) in AWS Organizations einzugrenzen.

John Stiles ist beispielsweise Administrator AWS Organizations . Er ist dafür verantwortlich, dass die Mitarbeiter des Unternehmens AWS-Konten nicht über zu viele Berechtigungen verfügen. Im Rahmen einer periodischen Sicherheitsprüfung überprüft er die Berechtigungen seiner Organisation. Seine `Development` Organisationseinheit enthält Konten, die häufig zum Testen neuer AWS Dienste verwendet werden. John überprüft den Bericht regelmäßig auf Services, auf die innerhalb der letzten 180 Tage nicht zugegriffen wurde. Anschließend entfernt er die Zugriffsberechtigungen auf diese Services für die Mitglieder der Organisationseinheit.

John meldet sich bei der IAM-Konsole mit seinen Anmeldeinformationen für das Hauptkonto an. In der IAM-Konsole sucht er die Organisationsdaten für die `Development`-Organisationseinheit. Er überprüft die Tabelle mit dem Bericht über den Dienstzugriff und sieht zwei AWS Dienste, auf die seit mehr als dem von ihm bevorzugten Zeitraum von 180 Tagen nicht zugegriffen wurde. Er erinnert sich, dass er den Entwicklungsteams Berechtigungen für den Zugriff auf Amazon Lex und hinzugefügt hat AWS Database Migration Service. John setzt sich mit den Entwicklungsteams in Verbindung und vergewissert sich, dass keine geschäftliche Notwendigkeit mehr besteht, um diese Services zu testen.

John ist jetzt bereit, auf der Grundlage der Informationen zum letzten Zugriff zu handeln. Er wählt Edit in AWS Organizations(In bearbeiten) aus und wird darauf hingewiesen, dass die SCP mehreren Elemente zugeordnet ist. Er wählt Continue (Fortfahren). Darin AWS Organizationsüberprüft er die Ziele, um zu erfahren, welchen Organizations der SCP zugeordnet ist. Alle Entitys befinden sich innerhalb der `Development`-Organisationseinheit.

John beschließt, den Zugriff auf Amazon Lex und AWS Database Migration Service Aktionen im `NewServiceTest` SCP zu verweigern. Durch diese Aktion wird der unnötige Zugriff auf die Services entfernt.

IAM-Aktion, zuletzt aufgerufene Informationsservices und Aktionen

In der folgenden Tabelle sind die AWS Dienste aufgeführt, für die Informationen zum letzten Zugriff der IAM-Aktion angezeigt werden. Eine Liste der Aktionen in den einzelnen Diensten finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste in der Service Authorization Reference.

Service	Servicepräfix
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Account Management	Konto
AWS Certificate Manager	acm
Amazon Managed Workflows für Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Cost Profiler für Anwendungen	application-cost-profiler
Einblicke in CloudWatch Amazon-Anwendungen	applicationinsights

Service	Servicepräfix
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	automatische Skalierung
AWS Marketplace	aws-marketplace
AWS Backup	backup
AWS Batch	Batch
Amazon Braket	braket
AWS Budgets	Budgets
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	clousearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact

Service	Servicepräfix
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Amazon-Rezensent	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Benachrichtigungen	codestar-notifications
Amazon Cognito-Identität	cognito-identity
Amazon-Cognito-Benutzerpools	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm

Service	Servicepräfix
DevOpsAmazon-Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Elastische Amazon DocumentDB-Cluster	docdb-elastic
AWS Directory Service	ds
Amazon-DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR auf EKS (EMR Container)	emr-containers

Service	Servicepräfix
Amazon EMR Serverless	emr-serverless
OpenSearch Amazon-Dienst	es
Amazon EventBridge	Veranstaltungen
Amazon CloudWatch offenbar	evidently
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam

Service	Servicepräfix
AWS Identitätsspeicher	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming für Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms

Service	Servicepräfix
AWS Lambda	Lambda
Amazon Lex	lex
AWS License Manager Linux-Abonnement-Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
CloudWatch Amazon-Protokolle	Protokolle
Amazon Lookout für Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB für Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Migration-Hub-Strategieempfehlungen	migrationhub-strategy

Service	Servicepräfix
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	Organisationen
AWS Panorama	panorama
AWS Performance Insights	pi
EventBridgeAmazon-Pfeifen	pipes
Amazon Polly	polly
Amazon Connect Customer Profiles	Profil
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Papierkorb	rbin
Amazon Relational Database Service	rds
Amazon-Redshift	redshift
Amazon Redshift-Daten-API	redshift-data

Service	Servicepräfix
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	Rekognition
AWS Resilience Hub	resiliencehub
AWS Ressourcen Explorer	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Rollen überall	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery-Kontrollen	route53-recovery-control-config
Amazon Route 53 Recovery-Bereitschaft	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatchRUM	rum
Amazon Simple Storage Service	S3
Amazon S3 in Outposts	s3-outposts
SageMaker Geospatial-Funktionen von Amazon	sagemaker-geospatial
Savings Plans	savingsplans
EventBridgeAmazon-Schemas	schemas
Amazon SimpleDB	sdb

Service	Servicepräfix
AWS Secrets Manager	secretsmanager
AWS Security Hub	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint-SMS- und Sprachnachrichten-Service	sms-voice
AWS Snowball	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
AWS Systems Manager für SAP	ssm-sap
AWS Step Functions	Status
AWS Security Token Service	sts

Service	Servicepräfix
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	Tag (Markierung)
Amazon Textract	textract/
Amazon Timestream	timestream
AWS Telco Network Builder	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	Übertragung
Amazon Translate	translate
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	Workspaces
AWS X-Ray	xray

Aktionen für die zuletzt aufgerufene Aktion

In der folgenden Tabelle sind die Aktionen aufgeführt, für die Informationen zur Aktion, auf die zuletzt zugegriffen wurde, verfügbar sind.

Servicepräfix	Aktionen
access-analyzer	Access-Analyzer: Regel ApplyArchive
	Access-Analyzer: Generierung CancelPolicy
	Access-Analyzer: CheckAccess NotGranted
	Zugriffsanalysator: CheckNo NewAccess
	access-analyzer: Vorschau CreateAccess
	Access-Analyzer: CreateAnalyzer
	access-analyzer: Regel CreateArchive
	Access-Analyzer: DeleteAnalyzer
	access-analyzer: Regel DeleteArchive
	access-analyzer: Vorschau GetAccess
	access-analyzer: Ressource GetAnalyzed
	Access-Analyzer: GetAnalyzer
	access-analyzer: Regel GetArchive
	Access-Analyzer: GetFinding
	access-analyzer: Richtlinie GetGenerated
	Access-Analyzer: ListAccess PreviewFindings
	Access-Analyzer: Vorschauen ListAccess
	access-analyzer: Ressourcen ListAnalyzed
	Access-Analyzer: ListAnalyzers
	access-analyzer: Regeln ListArchive
	Access-Analyzer: ListFindings

Servicepräfix	Aktionen
	Access-Analyzer: Generationen ListPolicy Access-Analyzer: Generierung StartPolicy Access-Analyzer: Scannen StartResource access-analyzer: Regel UpdateArchive Access-Analyzer: UpdateFindings Zugriffsanalysator: ValidatePolicy
Konto	Konto: Kontakt DeleteAlternate Konto: DisableRegion konto: EnableRegion Konto: GetAlternate Kontakt Konto: GetContact Informationen Konto: GetRegion OptStatus konto: ListRegions Konto: PutAlternate Kontakt Konto: PutContact Informationen

Servicepräfix	Aktionen
acm	acm: DeleteCertificate acm: DescribeCertificate acm: ExportCertificate acm: Konfiguration GetAccount acm: GetCertificate acm: ImportCertificate acm: ListCertificates acm: Konfiguration PutAccount acm: RenewCertificate acm: RequestCertificate acm: E-Mail ResendValidation acm: Optionen UpdateCertificate
airflow	Luftstrom: Token CreateCli Luftstrom: CreateEnvironment Luftstrom: CreateWeb LoginToken Luftstrom: DeleteEnvironment Luftstrom: GetEnvironment Luftstrom: ListEnvironments Luftstrom: PublishMetrics Luftstrom: UpdateEnvironment

Servicepräfix	Aktionen
amplify	verstärken: CreateApp
	verstärken: Umgebung CreateBackend
	verstärken: CreateBranch
	verstärken: CreateDeployment
	verstärken: Assoziation CreateDomain
	verstärken: Hook CreateWeb
	verstärken: DeleteApp
	verstärken: Umgebung DeleteBackend
	verstärken: DeleteBranch
	verstärken: Assoziation DeleteDomain
	verstärken: DeleteJob
	verstärken: Hook DeleteWeb
	verstärken: Logs GenerateAccess
	verstärken: GetApp
	verstärken: Url GetArtifact
	verstärken: Umgebung GetBackend
	verstärken: GetBranch
	verstärken: Assoziation GetDomain
	verstärken: GetJob
	verstärken: Hook GetWeb
	verstärken: ListApps

Servicepräfix	Aktionen
	verstärken: ListArtifacts
	verstärken: Umgebungen ListBackend
	verstärken: ListBranches
	verstärken: Assoziationen ListDomain
	verstärken: ListJobs
	verstärken: Hooks ListWeb
	verstärken: StartDeployment
	verstärken: StartJob
	verstärken: StopJob
	verstärken: UpdateApp
	verstärken: UpdateBranch
	verstärken: Assoziation UpdateDomain
	verstärken: Hook UpdateWeb

Servicepräfix	Aktionen
amplifyuibuilder	amplifiuibuilder: CreateComponent amplifiuibuilder: CreateForm amplifiuibuilder: CreateTheme amplifiuibuilder: DeleteComponent amplifiuibuilder: DeleteForm amplifiuibuilder: DeleteTheme amplifiuibuilder: ExportComponents amplifiuibuilder: ExportThemes amplifyuibuilder: Job GetCodegen amplifyuibuilder: Stellenangebote ListCodegen amplifyuibuilder: ListComponents amplifiuibuilder: ListForms amplifiuibuilder: ListThemes amplifyuibuilder: Flagge ResetMetadata amplifyuibuilder: Job StartCodegen amplifyuibuilder: UpdateComponent amplifiuibuilder: UpdateForm amplifiuibuilder: UpdateTheme

Servicepräfix	Aktionen
app-integrations	App-Integrationen: CreateApplication
	App-Integrationen: Integration CreateData
	App-Integrationen: Integration CreateEvent
	App-Integrationen: DeleteApplication
	App-Integrationen: Integration DeleteData
	App-Integrationen: Integration DeleteEvent
	App-Integrationen: GetApplication
	App-Integrationen: Integration GetData
	App-Integrationen: Integration GetEvent
	App-Integrationen: Verbände ListApplication
	App-Integrationen: ListApplications
	App-Integrationen: ListData IntegrationAssociations
	App-Integrationen: Integrationen ListData
	App-Integrationen: ListEvent IntegrationAssociations
	App-Integrationen: Integrationen ListEvent
	App-Integrationen: UpdateApplication
	App-Integrationen: Integration UpdateData
	App-Integrationen: Integration UpdateEvent

Servicepräfix	Aktionen
appconfig	Anwendungskonfiguration: CreateApplication appconfig: Profil CreateConfiguration appconfig: Strategie CreateDeployment appconfig: CreateEnvironment Anwendungskonfiguration: CreateExtension appconfig: Assoziation CreateExtension appconfig: CreateHosted ConfigurationVersion Anwendungskonfiguration: DeleteApplication appconfig: Profil DeleteConfiguration appconfig: Strategie DeleteDeployment appconfig: DeleteEnvironment Anwendungskonfiguration: DeleteExtension appconfig: Assoziation DeleteExtension appconfig: DeleteHosted ConfigurationVersion Anwendungskonfiguration: GetApplication Anwendungskonfiguration: GetConfiguration appconfig: Profil GetConfiguration appconfig: GetDeployment appconfig: Strategie GetDeployment appconfig: GetEnvironment Anwendungskonfiguration: GetExtension

Servicepräfix	Aktionen
	appconfig: Assoziation GetExtension appconfig: GetHosted ConfigurationVersion Anwendungskonfiguration: ListApplications appconfig: Profile ListConfiguration appconfig: ListDeployments appconfig: Strategien ListDeployment appconfig: ListEnvironments appconfig: Assoziationen ListExtension appconfig: ListExtensions Anwendungskonfiguration: ListHosted ConfigurationVersions Anwendungskonfiguration: StartDeployment Anwendungskonfiguration: StopDeployment Anwendungskonfiguration: UpdateApplication appconfig: Profil UpdateConfiguration appconfig: Strategie UpdateDeployment appconfig: UpdateEnvironment Anwendungskonfiguration: UpdateExtension appconfig: Assoziation UpdateExtension appconfig: ValidateConfiguration

Servicepräfix	Aktionen
appflow	appflow: Ausführungen CancelFlow appflow: Profil CreateConnector Appflow: CreateFlow appflow: Profil DeleteConnector Appflow: DeleteFlow Appflow: DescribeConnector appflow: Entität DescribeConnector appflow: Profile DescribeConnector Appflow: DescribeConnectors Appflow: DescribeFlow Appflow: DescribeFlow ExecutionRecords appflow: Entitäten ListConnector Appflow: ListConnectors Appflow: ListFlows Appflow: RegisterConnector Appflow: ResetConnector MetadataCache Appflow: StartFlow Appflow: StopFlow appflow: Konnektor UnRegister appflow: Profil UpdateConnector appflow: Registrierung UpdateConnector

Servicepräfix	Aktionen
	Appflow: UpdateFlow
application-cost-profiler	Anwendungskostenprofiler: Definition DeleteReport Anwendungskostenprofiler: GetReport Definition application-cost-profiler: ImportApplication Verwendung application-cost-profiler: ListReport Definitionen Anwendungskostenprofiler: PutReport Definition Anwendungskostenprofiler: UpdateReport Definition

Servicepräfix	Aktionen
applicationinsights	Einblicke in die Anwendung: AddWorkload
	Einblicke in die Anwendung: CreateApplication
	Einblicke in die Anwendung: CreateComponent
	Anwendungseinblicke: Muster CreateLog
	Einblicke in die Anwendung: DeleteApplication
	Einblicke in die Anwendung: DeleteComponent
	Anwendungseinblicke: Muster DeleteLog
	Einblicke in die Anwendung: DescribeApplication
	Einblicke in die Anwendung: DescribeComponent
	applicationinsights: Konfiguration DescribeComponent
	Einblicke in die Anwendung: DescribeComponent Configura tionRecommendation
	Anwendungseinblicke: Muster DescribeLog
	Einblicke in die Anwendung: DescribeObservation
	Einblicke in die Anwendung: DescribeProblem
	Anwendungseinblicke: Beobachtungen DescribeProblem
	Einblicke in die Anwendung: DescribeWorkload
	Einblicke in die Anwendung: ListApplications
	Einblicke in die Anwendung: ListComponents
	Einblicke in die Anwendung: Geschichte ListConfiguration
	Anwendungseinblicke: Muster ListLog

Servicepräfix	Aktionen
	Einblicke in die Anwendung: ListLog PatternSets
	Einblicke in die Anwendung: ListProblems
	Einblicke in die Anwendung: ListWorkloads
	Einblicke in die Anwendung: RemoveWorkload
	Einblicke in die Anwendung: UpdateApplication
	Einblicke in die Anwendung: UpdateComponent
	applicationinsights: Konfiguration UpdateComponent
	applicationinsights: Muster UpdateLog
	Einblicke in die Anwendung: UpdateWorkload

Servicepräfix	Aktionen
appmesh	appmesh: Route CreateGateway appmesh: CreateMesh Appmesh: CreateRoute appmesh: Gateway CreateVirtual appmesh: Knoten CreateVirtual appmesh: Router CreateVirtual appmesh: Dienst CreateVirtual appmesh: Route DeleteGateway appmesh: DeleteMesh Appmesh: DeleteRoute appmesh: Gateway DeleteVirtual appmesh: Knoten DeleteVirtual appmesh: Router DeleteVirtual appmesh: Dienst DeleteVirtual appmesh: Route DescribeGateway appmesh: DescribeMesh Appmesh: DescribeRoute appmesh: Gateway DescribeVirtual appmesh: Knoten DescribeVirtual appmesh: Router DescribeVirtual appmesh: Dienst DescribeVirtual

Servicepräfix	Aktionen
	appmesh: Routen ListGateway appmesh: ListMeshes Appmesh: ListRoutes appmesh: Gateways ListVirtual appmesh: Knoten ListVirtual appmesh: Router ListVirtual appmesh: Dienste ListVirtual appmesh: Ressourcen StreamAggregated appmesh: Route UpdateGateway appmesh: UpdateMesh Appmesh: UpdateRoute appmesh: Gateway UpdateVirtual appmesh: Knoten UpdateVirtual appmesh: Router UpdateVirtual appmesh: Dienst UpdateVirtual

Servicepräfix	Aktionen
appstream	Appstream: AssociateApp BlockBuilder AppBlock
	Appstream: Flotte AssociateApplication
	Appstream: AssociateApplication ToEntitlement
	Appstream: AssociateFleet
	Appstream: BatchAssociate UserStack
	Appstream: BatchDisassociate UserStack
	Appstream: CopyImage
	Appstream: Blockieren CreateApp
	Appstream: CreateApp BlockBuilder
	Appstream: Streaming-URL CreateApp BlockBuilder
	Appstream: CreateApplication
	Appstream: Config CreateDirectory
	Appstream: CreateEntitlement
	Appstream: CreateFleet
	Appstream: Builder CreateImage
	Appstream: URL CreateImage BuilderStreaming
	Appstream: CreateStack
	Appstream: URL CreateStreaming
	Appstream: Bild CreateUpdated
	Appstream: CreateUsage ReportSubscription
	Appstream: CreateUser

Servicepräfix	Aktionen
	Appstream: Blockieren DeleteApp
	Appstream: DeleteApp BlockBuilder
	Appstream: DeleteApplication
	Appstream: Config DeleteDirectory
	Appstream: DeleteEntitlement
	Appstream: DeleteFleet
	Appstream: DeleteImage
	Appstream: Builder DeleteImage
	appstream: Berechtigungen DeleteImage
	Appstream: DeleteStack
	Appstream: DeleteUsage ReportSubscription
	Appstream: DeleteUser
	appstream: Verbände DescribeApp BlockBuilder AppBlock
	Appstream: DescribeApp BlockBuilders
	appstream: Blöcke DescribeApp
	Appstream: DescribeApplication FleetAssociations
	Appstream: DescribeApplications
	Appstream: Konfigurationen DescribeDirectory
	Appstream: DescribeEntitlements
	Appstream: DescribeFleets
	Appstream: Entwickler DescribeImage

Servicepräfix	Aktionen
	appstream: Berechtigungen DescribeImage
	Appstream: DescribeImages
	Appstream: DescribeSessions
	Appstream: DescribeStacks
	Appstream: DescribeUsage ReportSubscriptions
	Appstream: DescribeUsers
	Appstream: DescribeUser StackAssociations
	Appstream: DisableUser
	Appstream: DisassociateApp BlockBuilder AppBlock
	Appstream: Flotte DisassociateApplication
	Appstream: DisassociateApplication FromEntitlement
	Appstream: DisassociateFleet
	Appstream: EnableUser
	Appstream: ExpireSession
	Appstream: ListAssociated Flotten
	Appstream: Stapel ListAssociated
	appstream: ListEntitled Anwendungen
	Appstream: StartApp BlockBuilder
	Appstream: StartFleet
	Appstream: Builder StartImage
	Appstream: StopApp BlockBuilder

Servicepräfix	Aktionen
	Appstream: StopFleet
	Appstream: Builder StopImage
	Appstream: UpdateApp BlockBuilder
	Appstream: UpdateApplication
	Appstream: Config UpdateDirectory
	Appstream: UpdateEntitlement
	Appstream: UpdateFleet
	appstream: Berechtigungen UpdateImage
	Appstream: UpdateStack

Servicepräfix	Aktionen
appsync	App-Synchronisierung: AssociateApi AppSync: AssociateMerged GraphqlApi AppSync: AssociateSource GraphqlApi appsync: Zwischenspeicher CreateApi appsync: Schlüssel CreateApi appsync: Quelle CreateData appsync: Name CreateDomain Appsync: CreateFunction Appsync: Api CreateGraphql AppSync: CreateResolver AppSync: CreateType appsync: Zwischenspeicher DeleteApi appsync: Schlüssel DeleteApi appsync: Quelle DeleteData appsync: Name DeleteDomain Appsync: DeleteFunction Appsync: Api DeleteGraphql AppSync: DeleteResolver AppSync: DeleteType AppSync: DisassociateApi AppSync: DisassociateMerged GraphqlApi

Servicepräfix	Aktionen
	AppSync: DisassociateSource GraphqlApi AppSync: EvaluateCode appsync: Vorlage EvaluateMapping appsync: Cache FlushApi appsync: Assoziation GetApi appsync: Cache GetApi appsync: Quelle GetData Appsync: GetData SourceIntrospection appsync: Name GetDomain Appsync: GetFunction Appsync: Api GetGraphql appsync: Variablen GetGraphql ApiEnvironment appsync: Schema GetIntrospection Appsync: GetResolver AppSync: GetSchema CreationStatus AppSync: GetSource ApiAssociation AppSync: GetType appsync: Schlüssel ListApi appsync: Quellen ListData appsync: Namen ListDomain appsync: ListFunctions

Servicepräfix	Aktionen
	appsync: Apis ListGraphql Appsync: ListResolvers AppSync: ListResolvers ByFunction AppSync: ListSource ApiAssociations AppSync: ListTypes AppSync: ListTypes ByAssociation appsync: Variablen PutGraphql ApiEnvironment Appsync: StartData SourceIntrospection appsync: Erstellung StartSchema appsync: Zusammenführen StartSchema appsync: Zwischenspeicher UpdateApi appsync: Schlüssel UpdateApi appsync: Quelle UpdateData appsync: Name UpdateDomain Appsync: UpdateFunction Appsync: Api UpdateGraphql AppSync: UpdateResolver AppSync: UpdateSource ApiAssociation AppSync: UpdateType

Servicepräfix	Aktionen
aps	Apps: CreateAlert ManagerDefinition
	aps: CreateLogging Konfiguration
	Apps: CreateRule GroupsNamespace
	Apps: CreateScraper
	Apps: CreateWorkspace
	Apps: DeleteAlert ManagerDefinition
	aps: DeleteLogging Konfiguration
	Apps: DeleteRule GroupsNamespace
	Apps: DeleteScraper
	Apps: DeleteWorkspace
	Apps: DescribeAlert ManagerDefinition
	aps: DescribeLogging Konfiguration
	Apps: DescribeRule GroupsNamespace
	Apps: DescribeScraper
	Apps: DescribeWorkspace
	Apps: GetDefault ScraperConfiguration
	Apps: ListRule GroupsNamespaces
	Apps: ListScrapers
	Apps: ListWorkspaces
	Apps: PutAlert ManagerDefinition
	Apps: PutRule GroupsNamespace

Servicepräfix	Aktionen
	aps: UpdateLogging Konfiguration aps: UpdateWorkspace Alias

Servicepräfix	Aktionen
athena	Athene: BatchGet NamedQuery Athene: BatchGet PreparedStatement Athene: BatchGet QueryExecution Athena: Reservierung CancelCapacity Athena: Reservierung CreateCapacity athena: Katalog CreateData Athena: Anfrage CreateNamed Athena: CreateNotebook Athena: Aussage CreatePrepared Athena: CreatePresigned NotebookUrl Athena: Gruppe CreateWork Athena: Reservierung DeleteCapacity athena: Katalog DeleteData Athena: Anfrage DeleteNamed Athena: DeleteNotebook Athena: Aussage DeletePrepared Athena: Gruppe DeleteWork Athena: ExportNotebook Athena: Hinrichtung GetCalculation Athene: GetCalculation ExecutionCode Athene: GetCalculation ExecutionStatus

Servicepräfix	Aktionen
	Athene: GetCapacity AssignmentConfiguration Athena: Reservierung GetCapacity Athene: GetDatabase athena: Katalog GetData Athena: Anfrage GetNamed athena: Metadaten GetNotebook athena: Aussage GetPrepared Athena: Hinrichtung GetQuery Athena: Ergebnisse GetQuery Athena: GetQuery ResultsStream Athene: GetQuery RuntimeStatistics Athene: GetSession Athena: Status GetSession athena: Metadaten GetTable Athena: Gruppe GetWork Athena: ImportNotebook Athena: DPU-Größen ListApplication Athena: Hinrichtungen ListCalculation Athena: Reservierungen ListCapacity Athena: ListDatabases athena: Kataloge ListData

Servicepräfix	Aktionen
	athena: Versionen ListEngine Athena: ListExecutors athena: Abfragen ListNamed athena: Metadaten ListNotebook Athena: Sitzungen ListNotebook athena: Aussagen ListPrepared Athena: Hinrichtungen ListQuery Athena: ListSessions athena: Metadaten ListTable athena: Gruppen ListWork Athena: PutCapacity AssignmentConfiguration Athena: Hinrichtung StartCalculation Athena: Hinrichtung StartQuery Athene: StartSession Athena: Hinrichtung StopCalculation Athena: Hinrichtung StopQuery Athene: TerminateSession Athena: Reservierung UpdateCapacity athena: Katalog UpdateData Athena: Anfrage UpdateNamed Athena: UpdateNotebook

Servicepräfix	Aktionen
	athena: Metadaten UpdateNotebook athena: Aussage UpdatePrepared Athena: Gruppe UpdateWork

Servicepräfix	Aktionen
auditmanager	auditmanager: Ordner AssociateAssessment ReportEvidence
	auditmanager: Beweise BatchAssociate AssessmentReport
	Auditmanager: Bewertung BatchCreate DelegationBy
	auditmanager: Bewertung BatchDelete DelegationBy
	Auditmanager: Beweise BatchDisassociate AssessmentReport
	Prüfungsleiter: BatchImport EvidenceTo AssessmentControl
	Prüfungsleiter: CreateAssessment
	Auditmanager: Framework CreateAssessment
	auditmanager: Bericht CreateAssessment
	Prüfungsleiter: CreateControl
	Prüfungsleiter: DeleteAssessment
	Auditmanager: Framework DeleteAssessment
	Prüfungsleiter: DeleteAssessment FrameworkShare
	auditmanager: Bericht DeleteAssessment
	Prüfungsleiter: DeleteControl
	Prüfungsleiter: DeregisterAccount
	Prüfungsleiter: DeregisterOrganization AdminAccount
	auditmanager: Ordner DisassociateAssessment ReportEvidence
	auditmanager: Status GetAccount
	Prüfungsleiter: GetAssessment
	Auditmanager: Framework GetAssessment

Servicepräfix	Aktionen
	Prüfungsleiter: GetAssessment ReportUrl
	auditmanager: Protokolle GetChange
	Auditmanager: GetControl
	Prüfungsleiter: GetDelegations
	Prüfungsleiter: GetEvidence
	auditmanager: Ordner GetEvidence ByEvidence
	auditmanager: URL GetEvidence FileUpload
	auditmanager: Ordner GetEvidence
	auditmanager: Bewertung GetEvidence FoldersBy
	Prüfungsleiter: GetEvidence FoldersBy AssessmentControl
	Prüfungsleiter: GetInsights
	Prüfungsleiter: GetInsights ByAssessment
	Prüfungsleiter: GetOrganization AdminAccount
	Prüfungsleiter: GetServices InScope
	Prüfungsleiter: GetSettings
	auditmanager: Domäne ListAssessment ControlInsights ByControl
	auditmanager: Frameworks ListAssessment
	auditmanager: Anfragen ListAssessment FrameworkShare
	auditmanager: Berichte ListAssessment
	Auditmanager: ListAssessments
	Prüfungsleiter: ListControl DomainInsights

Servicepräfix	Aktionen
	Prüfungsleiter: ListControl DomainInsights ByAssessment
	Prüfungsleiter: ListControl InsightsBy ControlDomain
	Prüfungsleiter: ListControls
	auditmanager: Quelle ListKeywords ForData
	auditmanager: ListNotifications
	Prüfungsleiter: RegisterAccount
	Prüfungsleiter: RegisterOrganization AdminAccount
	Prüfungsleiter: StartAssessment FrameworkShare
	Prüfungsleiter: UpdateAssessment
	auditmanager: Steuerung UpdateAssessment
	auditmanager: Status UpdateAssessment ControlSet
	Auditmanager: Framework UpdateAssessment
	Prüfungsleiter: UpdateAssessment FrameworkShare
	Prüfungsleiter: Status UpdateAssessment
	Prüfungsleiter: UpdateControl
	Prüfungsleiter: UpdateSettings
	Prüfungsleiter: ValidateAssessment ReportIntegrity

Servicepräfix	Aktionen
automatische Skalierung	automatische Skalierung: AttachInstances automatische Skalierung: Balancer AttachLoad Autoscaling: Gruppen AttachLoad BalancerTarget Autoscaling: Quellen AttachTraffic automatische Skalierung: BatchDelete ScheduledAction automatische Skalierung: BatchPut ScheduledUpdate GroupAction Autoscaling: Aktualisieren CancelInstance Autoscaling: Aktion CompleteLifecycle automatische Skalierung: CreateAuto ScalingGroup Autoscaling: Konfiguration CreateLaunch automatische Skalierung: DeleteAuto ScalingGroup Autoscaling: Konfiguration DeleteLaunch Autoscaling: Hook DeleteLifecycle Autoscaling: Konfiguration DeleteNotification automatische Skalierung: DeletePolicy Autoscaling: Aktion DeleteScheduled automatische Skalierung: Pool DeleteWarm Autoscaling: Grenzwerte DescribeAccount Autoscaling: Typen DescribeAdjustment automatische Skalierung: DescribeAuto ScalingGroups automatische Skalierung: DescribeAuto ScalingInstances

Servicepräfix	Aktionen
	Autoscaling: Typen DescribeAuto ScalingNotification Autoscaling: Wird aktualisiert DescribeInstance Autoscaling: Konfigurationen DescribeLaunch Autoscaling: Hooks DescribeLifecycle automatische Skalierung: DescribeLifecycle HookTypes automatische Skalierung: Balancer DescribeLoad Autoscaling: Gruppen DescribeLoad BalancerTarget automatische Skalierung: DescribeMetric CollectionTypes Autoscaling: Konfigurationen DescribeNotification automatische Skalierung: DescribePolicies Autoscaling: Aktivitäten DescribeScaling automatische Skalierung: DescribeScaling ProcessTypes Autoscaling: Aktionen DescribeScheduled automatische Skalierung: DescribeTermination PolicyTypes Autoscaling: Quellen DescribeTraffic automatische Skalierung: Pool DescribeWarm automatische Skalierung: DetachInstances automatische Skalierung: Balancer DetachLoad Autoscaling: Gruppen DetachLoad BalancerTarget Autoscaling: Quellen DetachTraffic Autoscaling: Sammlung DisableMetrics

Servicepräfix	Aktionen
	Autoscaling: Sammlung EnableMetrics automatische Skalierung: EnterStandby automatische Skalierung: ExecutePolicy automatische Skalierung: ExitStandby automatische Skalierung: GetPredictive ScalingForecast automatische Skalierung: Hook PutLifecycle Autoscaling: Konfiguration PutNotification Autoscaling: Richtlinie PutScaling Autoscaling: Aktion PutScheduled UpdateGroup automatische Skalierung: Pool PutWarm automatische Skalierung: RecordLifecycle ActionHeartbeat automatische Skalierung: ResumeProcesses Autoscaling: Aktualisieren RollbackInstance Autoscaling: Kapazität SetDesired Autoscaling: Health SetInstance Autoscaling: Schutz SetInstance Autoscaling: Aktualisieren StartInstance automatische Skalierung: SuspendProcesses automatische Skalierung: TerminateInstance InAuto ScalingGroup automatische Skalierung: UpdateAuto ScalingGroup
aws-marketplace	AWS-Marktplatz: GetEntitlements

Servicepräfix	Aktionen
backup	Sicherung: Halten CancelLegal Backup: CreateBackup Plan Backup: CreateBackup Auswahl Sicherung: CreateBackup Tresor Sicherung: CreateFramework Sicherung: CreateLegal Halten Sicherung: CreateLogically AirGapped BackupVault Sicherung: CreateReport Plan Sicherungskopie: CreateRestore TestingPlan Sicherungskopie: CreateRestore TestingSelection Sicherung: DeleteBackup Plan Backup: DeleteBackup Auswahl Sicherung: DeleteBackup Tresor Backup: DeleteBackup VaultAccess Richtlinie Backup: DeleteBackup VaultLock Konfiguration Sicherung: DeleteBackup VaultNotifications Sicherung: DeleteFramework Backup: DeleteRecovery Punkt Backup: DeleteReport Plan Sicherungskopie: DeleteRestore TestingPlan Sicherungskopie: DeleteRestore TestingSelection

Servicepräfix	Aktionen
	Sicherungskopie: DescribeBackup Job Sicherung: DescribeBackup Tresor Sicherungskopie: DescribeCopy Job Sicherung: DescribeFramework Backup: DescribeGlobal Einstellungen Backup: DescribeProtected Ressource Backup: DescribeRecovery Punkt Backup: DescribeRegion Einstellungen Sicherungskopie: DescribeReport Job Backup: DescribeReport Plan Sicherungskopie: DescribeRestore Job Backup: DisassociateRecovery Punkt Backup: DisassociateRecovery PointFrom Elternteil Sicherungskopie: ExportBackup PlanTemplate Sicherung: GetBackup Plan Sicherung: GetBackup PlanFrom JSON Sicherung: GetBackup PlanFrom Vorlage Backup: GetBackup Auswahl Backup: GetBackup VaultAccess Richtlinie Sicherung: GetBackup VaultNotifications Sicherung: GetLegal Halten

Servicepräfix	Aktionen
	Backup: GetRecovery PointRestore Metadaten Sicherung: GetRestore JobMetadata Sicherung: GetRestore TestingInferred Metadaten Sicherung: GetRestore TestingPlan Sicherungskopie: GetRestore TestingSelection Sicherungskopie: GetSupported ResourceTypes Backup: ListBackup Jobs Sicherung: ListBackup JobSummaries Backup: ListBackup Pläne Sicherung: ListBackup PlanTemplates Sicherungskopie: ListBackup PlanVersions Backup: ListBackup Auswahlen Backup: ListBackup Tresore Backup: Jobs ListCopy Sicherung: ListCopy JobSummaries Sicherung: ListFrameworks Sicherung: ListLegal Hält Backup: ListProtected Ressourcen Sicherung: ListRecovery PointsBy BackupVault Sicherungskopie: ListRecovery PointsBy LegalHold Backup: ListRecovery PointsBy Ressource

Servicepräfix	Aktionen
	Backup: ListReport Jobs Backup: ListReport Pläne Backup: ListRestore Jobs Sicherung: ListRestore JobsBy ProtectedResource Sicherungskopie: ListRestore JobSummaries Sicherungskopie: ListRestore TestingPlans Sicherungskopie: ListRestore TestingSelections Backup: PutBackup VaultAccess Richtlinie Backup: PutBackup VaultLock Konfiguration Sicherung: PutBackup VaultNotifications Sicherungskopie: PutRestore ValidationResult Sicherungskopie: StartBackup Job Sicherungskopie: StartCopy Job Sicherungskopie: StartReport Job Sicherungskopie: StartRestore Job Sicherungskopie: StopBackup Job Backup: UpdateBackup Plan Sicherungskopie: UpdateFramework Backup: UpdateGlobal Einstellungen Sicherungskopie: UpdateRecovery PointLifecycle Backup: UpdateRegion Einstellungen

Servicepräfix	Aktionen
	Backup: UpdateReport Plan Sicherungskopie: UpdateRestore TestingPlan Sicherungskopie: UpdateRestore TestingSelection

Servicepräfix	Aktionen
Batch	Stapel: CancelJob Batch: CreateCompute Umwelt Stapel: CreateJob Warteschlange Stapel: CreateScheduling Richtlinie Batch: DeleteCompute Umgebung Stapel: DeleteJob Warteschlange Stapel: DeleteScheduling Richtlinie Stapel: DeregisterJob Definition Batch: DescribeCompute Umgebungen Batch: DescribeJob Definitionen Stapel: DescribeJob Warteschlangen Stapel: DescribeJobs batch: DescribeScheduling Richtlinien Stapel: ListJobs batch: ListScheduling Richtlinien Stapel: RegisterJob Definition Stapel: SubmitJob Charge: TerminateJob Batch: UpdateCompute Umwelt Stapel: UpdateJob Warteschlange Stapel: UpdateScheduling Richtlinie

Servicepräfix	Aktionen
braket	Klammer: Vereinbarung AcceptUser Klammer: Merkmal AccessBraket Halterung: CancelJob Klammer: Aufgabe CancelQuantum Klammer: CreateJob Klammer: Aufgabe CreateQuantum Klammer: GetDevice Klammer: GetJob Klammer: Aufgabe GetQuantum Klammer: Status GetService LinkedRole Halterung: GetUser AgreementStatus Klammer: SearchDevices Klammer: SearchJobs Klammer: Aufgaben SearchQuantum

Servicepräfix	Aktionen
Budgets	Budgets: ModifyBudget
	Budgets: CreateBudget Maßnahmen
	Budgets: ModifyBudget
	Budgets: ModifyBudget
	Budgets: ModifyBudget
	Budgets: DeleteBudget Maßnahmen
	Budgets: ModifyBudget
	Budgets: ModifyBudget
	Budgets: ViewBudget
	Budgets: DescribeBudget Maßnahmen
	Budgets: DescribeBudget ActionHistories
	Budgets: DescribeBudget ActionsFor Konto
	Budgets: DescribeBudget ActionsFor Budget
	Budgets: ViewBudget
	Budgets: ViewBudget
	Budgets: ViewBudget
	Budgets: ViewBudget
	Budgets: ViewBudget
	Budgets: ExecuteBudget Maßnahmen
	Budgets: ModifyBudget
	Budgets: UpdateBudget Maßnahmen

Servicepräfix	Aktionen
	Budgets: ModifyBudget Budgets: ModifyBudget
cloud9	wolke9: EC2 CreateEnvironment cloud9: Mitgliedschaft CreateEnvironment cloud9: DeleteEnvironment cloud9: Mitgliedschaft DeleteEnvironment cloud9: Mitgliedschaften DescribeEnvironment cloud9: DescribeEnvironments wolke9: Status DescribeEnvironment Wolke 9: ListEnvironments Wolke 9: UpdateEnvironment cloud9: Mitgliedschaft UpdateEnvironment

Servicepräfix	Aktionen
cloudformation	Cloud-Formation: BatchDescribe TypeConfigurations
	Wolkenbildung: Stapel CancelUpdate
	Cloudformation: Rollback ContinueUpdate
	Cloudformation: Eingestellt CreateChange
	Cloudformation: Vorlage CreateGenerated
	Wolkenbildung: CreateStack
	Cloudformation: Instanzen CreateStack
	Cloudformation: Satz CreateStack
	Wolkenbildung: DeactivateType
	Wolkenbildung: Eingestellt DeleteChange
	Cloudformation: Vorlage DeleteGenerated
	Wolkenbildung: DeleteStack
	Cloudformation: Instanzen DeleteStack
	Cloudformation: Satz DeleteStack
	Wolkenbildung: DeregisterType
	Wolkenbildung: Grenzen DescribeAccount
	Cloudformation: Eingestellt DescribeChange
	Wolkenbildung: DescribeChange SetHooks
	Cloudformation: Vorlage DescribeGenerated
	cloudformation: Zugriff DescribeOrganizations
	Wolkenbildung: DescribePublisher

Servicepräfix	Aktionen
	Wolkenbildung: Scannen DescribeResource
	Cloudformation: Status DescribeStack DriftDetection
	Cloudformation: Ereignisse DescribeStack
	cloudformation: Instanz DescribeStack
	cloudformation: Ressource DescribeStack
	Wolkenbildung: DescribeStack ResourceDrifts
	Cloudformation: Ressourcen DescribeStack
	Wolkenbildung: DescribeStacks
	Wolkenbildung: Eingestellt DescribeStack
	Wolkenbildung: DescribeStack SetOperation
	Wolkenbildung: DescribeType
	cloudformation: Registrierung DescribeType
	Cloudformation: Drift DetectStack
	Wolkenbildung: DetectStack ResourceDrift
	Wolkenbildung: DetectStack SetDrift
	Cloudformation: Kosten EstimateTemplate
	Cloudformation: Set ExecuteChange
	Cloudformation: Vorlage GetGenerated
	Cloudformation: Richtlinie GetStack
	Cloudformation: GetTemplate
	Cloudformation: Zusammenfassung GetTemplate

Servicepräfix	Aktionen
	Cloudformation: Satz ImportStacks ToStack
	Cloudformation: Sätze ListChange
	Wolkenbildung: ListExports
	Cloudformation: Vorlagen ListGenerated
	Wolkenbildung: ListImports
	Cloudformation: Ressourcen ListResource ScanRelated
	Wolkenbildung: ListResource ScanResources
	Wolkenbildung: Scans ListResource
	Wolkenbildung: Drifts ListStack InstanceResource
	Cloudformation: Instanzen ListStack
	Cloudformation: Ressourcen ListStack
	Wolkenbildung: ListStack SetAuto DeploymentTargets
	Wolkenbildung: Ergebnisse ListStack SetOperation
	Wolkenbildung: ListStack SetOperations
	Wolkenbildung: Sätze ListStack
	cloudformation: Anmeldungen ListType
	Cloudformation: ListTypes
	Cloudformation: Versionen ListType
	Wolkenbildung: PublishType
	Wolkenbildung: Fortschritte RecordHandler
	Wolkenbildung: RegisterPublisher

Servicepräfix	Aktionen
	Wolkenbildung: RegisterType
	Wolkenbildung: RollbackStack
	Cloudformation: Richtlinie SetStack
	cloudformation: Konfiguration SetType
	Cloudformation: SetType DefaultVersion
	Wolkenbildung: SignalResource
	Wolkenbildung: Scannen StartResource
	Wolkenbildung: StopStack SetOperation
	Wolkenbildung: TestType
	Cloudformation: Vorlage UpdateGenerated
	Wolkenbildung: UpdateStack
	Cloudformation: Instanzen UpdateStack
	Cloudformation: Satz UpdateStack
	Cloudformation: Schutz UpdateTermination
	Wolkenbildung: ValidateTemplate

Servicepräfix	Aktionen
cloudfront	Wolkenfront: AssociateAlias Cloudfront: Richtlinie CreateCache Cloudfront: CreateCloud FrontOrigin AccessIdentity Wolkenfront: CreateContinuous DeploymentPolicy Cloudfront: Config CreateField LevelEncryption cloudfront: Profil CreateField LevelEncryption Cloudfront: CreateFunction Wolkenfront: CreateInvalidation cloudfront: Gruppe CreateKey Cloudfront: CreateKey ValueStore cloudfront: Abonnement CreateMonitoring Cloudfront: CreateOrigin AccessControl Wolkenfront: CreateOrigin RequestPolicy Cloudfront: Schlüssel CreatePublic Wolkenfront: CreateRealtime LogConfig Wolkenfront: CreateResponse HeadersPolicy Cloudfront: Richtlinie DeleteCache Cloudfront: DeleteCloud FrontOrigin AccessIdentity Wolkenfront: DeleteContinuous DeploymentPolicy Wolkenfront: DeleteDistribution Cloudfront: Config DeleteField LevelEncryption

Servicepräfix	Aktionen
	cloudfront: Profil DeleteField LevelEncryption Cloudfront: DeleteFunction cloudfront: Gruppe DeleteKey Cloudfront: DeleteKey ValueStore cloudfront: Abonnement DeleteMonitoring Cloudfront: DeleteOrigin AccessControl Wolkenfront: DeleteOrigin RequestPolicy Cloudfront: Schlüssel DeletePublic Wolkenfront: DeleteRealtime LogConfig Wolkenfront: DeleteResponse HeadersPolicy Cloudfront: Vertrieb DeleteStreaming Wolkenfront: DescribeFunction Wolkenfront: DescribeKey ValueStore Cloudfront: Richtlinie GetCache Cloudfront: GetCache PolicyConfig Wolkenfront: GetCloud FrontOrigin AccessIdentity Cloudfront: Config GetCloud FrontOrigin AccessIdentity Cloudfront: GetContinuous DeploymentPolicy Cloudfront: Config GetContinuous DeploymentPolicy Cloudfront: Config GetDistribution Cloudfront: GetField LevelEncryption

Servicepräfix	Aktionen
	Cloudfront: Config GetField LevelEncryption cloudfront: Profil GetField LevelEncryption Cloudfront: GetField LevelEncryption ProfileConfig Wolkenfront: GetFunction Wolkenfront: GetInvalidation cloudfront: Gruppe GetKey Cloudfront: GetKey GroupConfig cloudfront: Abonnement GetMonitoring Cloudfront: GetOrigin AccessControl Cloudfront: Config GetOrigin AccessControl Cloudfront: GetOrigin RequestPolicy Cloudfront: Config GetOrigin RequestPolicy cloudfront: Schlüssel GetPublic Wolkenfront: GetPublic KeyConfig Wolkenfront: GetRealtime LogConfig Wolkenfront: GetResponse HeadersPolicy Cloudfront: Config GetResponse HeadersPolicy Cloudfront: Vertrieb GetStreaming Wolkenfront: GetStreaming DistributionConfig Cloudfront: Richtlinien ListCache Cloudfront: ListCloud FrontOrigin AccessIdentities

Servicepräfix	Aktionen
	Cloudfront: Aliase ListConflicting Cloudfront: ListContinuous DeploymentPolicies Wolkenfront: ListDistributions Wolkenfront: ListDistributions ByCache PolicyId cloudfront: Gruppe ListDistributions ByKey Cloudfront: ID ListDistributions ByOrigin RequestPolicy Wolkenfront: ListDistributions ByRealtime LogConfig Wolkenfront: ID ListDistributions ByResponse HeadersPolicy Cloudfront: Gültig ListDistributions ByWeb Cloudfront: Konfigurationen ListField LevelEncryption Cloudfront: Profile ListField LevelEncryption Cloudfront: ListFunctions Wolkenfront: ListInvalidations cloudfront: Gruppen ListKey Cloudfront: ListKey ValueStores Wolkenfront: ListOrigin AccessControls Wolkenfront: ListOrigin RequestPolicies Cloudfront: Schlüssel ListPublic Wolkenfront: ListRealtime LogConfigs Wolkenfront: ListResponse HeadersPolicies Cloudfront: Verteilungen ListStreaming

Servicepräfix	Aktionen
	Cloudfront: PublishFunction Wolkenfront: TestFunction Cloudfront: Richtlinie UpdateCache Cloudfront: UpdateCloud FrontOrigin AccessIdentity Wolkenfront: UpdateContinuous DeploymentPolicy Wolkenfront: UpdateDistribution Cloudfront: Config UpdateField LevelEncryption cloudfront: Profil UpdateField LevelEncryption Cloudfront: UpdateFunction cloudfront: Gruppe UpdateKey Cloudfront: UpdateKey ValueStore Wolkenfront: UpdateOrigin AccessControl Wolkenfront: UpdateOrigin RequestPolicy Cloudfront: Schlüssel UpdatePublic Wolkenfront: UpdateRealtime LogConfig Wolkenfront: UpdateResponse HeadersPolicy

Servicepräfix	Aktionen
cloudhsm	Wolkenbruch: CreateHapg
	cloudhsm: CreateHsm
	cloudhsm: Kunde CreateLuna
	cloudhsm: DeleteBackup
	cloudhsm: DeleteHapg
	cloudhsm: DeleteHsm
	cloudhsm: Kunde DeleteLuna
	cloudhsm: DescribeBackups
	cloudhsm: DescribeClusters
	cloudhsm: DescribeHapg
	cloudhsm: DescribeHsm
	cloudhsm: Kunde DescribeLuna
	cloudhsm: GetConfig
	cloudhsm: InitializeCluster
	cloudhsm: Zonen ListAvailable
	cloudhsm: ListHapgs
	cloudhsm: ListHsms
	cloudhsm: Kunden ListLuna
	cloudhsm: Attribute ModifyBackup
	cloudhsm: ModifyCluster
	cloudhsm: ModifyHapg

Servicepräfix	Aktionen
	cloudhsm: ModifyHsm cloudhsm: Kunde ModifyLuna cloudhsm: RestoreBackup

Servicepräfix	Aktionen
clousearch	Cloud-Suche: BuildSuggesters Cloud-Suche: CreateDomain cloudsearch: Schema DefineAnalysis Cloudsearch: DefineExpression cloudsearch: Feld DefineIndex Cloudsearch: DefineSuggester cloudsearch: Schema DeleteAnalysis Cloudsearch: DeleteDomain Cloud-Suche: DeleteExpression cloudsearch: Feld DeleteIndex Cloudsearch: DeleteSuggester cloudsearch: Schemata DescribeAnalysis cloudsearch: Optionen DescribeAvailability Cloudsearch: DescribeDomain EndpointOptions Cloud-Suche: DescribeDomains Cloud-Suche: DescribeExpressions cloudsearch: Felder DescribeIndex cloudsearch: Parameter DescribeScaling Cloudsearch: DescribeService AccessPolicies Cloud-Suche: DescribeSuggesters Cloud-Suche: IndexDocuments

Servicepräfix	Aktionen
	cloudsearch: Namen ListDomain cloudsearch: Optionen UpdateAvailability Cloudsearch: UpdateDomain EndpointOptions cloudsearch: Parameter UpdateScaling Cloudsearch: UpdateService AccessPolicies

Servicepräfix	Aktionen
cloudtrail	Wolkenspur: CancelQuery
	Wolkenspur: CreateChannel
	Wolkenspur: CreateEvent DataStore
	Wolkenspur: CreateTrail
	Wolkenspur: DeleteChannel
	Wolkenspur: DeleteEvent DataStore
	cloudtrail: Richtlinie DeleteResource
	cloudtrail: DeleteTrail
	Wolkenspur: DeregisterOrganization DelegatedAdmin
	Wolkenspur: DescribeQuery
	Wolkenspur: DescribeTrails
	Wolkenspur: DisableFederation
	Wolkenspur: GetChannel
	Wolkenspur: GetEvent DataStore
	cloudtrail: Daten GetEvent DataStore
	cloudtrail: Selektoren GetEvent
	Cloudtrail: GetImport
	cloudtrail: Selektoren GetInsight
	cloudtrail: Ergebnisse GetQuery
	cloudtrail: Richtlinie GetResource
	cloudtrail: GetTrail

Servicepräfix	Aktionen
	cloudtrail: Status GetTrail
	Cloudtrail: ListChannels
	Wolkenspur: ListEvent DataStores
	cloudtrail: Ausfälle ListImport
	cloudtrail: ListImports
	cloudtrail: Schlüssel ListPublic
	Cloudtrail: ListQueries
	Wolkenspur: ListTrails
	Wolkenspur: LookupEvents
	cloudtrail: Selektoren PutEvent
	cloudtrail: Selektoren PutInsight
	cloudtrail: Richtlinie PutResource
	cloudtrail: RegisterOrganization DelegatedAdmin
	Wolkenspur: RestoreEvent DataStore
	cloudtrail: Verschlucken StartEvent DataStore
	Cloudtrail: StartImport
	Wolkenspur: StartLogging
	Wolkenspur: StartQuery
	cloudtrail: Verschlucken StopEvent DataStore
	Cloudtrail: StopImport
	Wolkenspur: StopLogging

Servicepräfix	Aktionen
	Wolkenspur: UpdateChannel Wolkenspur: UpdateEvent DataStore Wolkenspur: UpdateTrail

Servicepräfix	Aktionen
cloudwatch	Wolkenbeobachtung: DeleteAlarms cloudwatch: Detektor DeleteAnomaly Cloudwatch: DeleteDashboards cloudwatch: Regeln DeleteInsight cloudwatch: Streamen DeleteMetric cloudwatch: Geschichte DescribeAlarm Cloudwatch: DescribeAlarms Cloudwatch: DescribeAlarms ForMetric cloudwatch: Detektoren DescribeAnomaly cloudwatch: Regeln DescribeInsight cloudwatch: Aktionen DisableAlarm cloudwatch: Regeln DisableInsight cloudwatch: Aktionen EnableAlarm cloudwatch: Regeln EnableInsight Cloudwatch: GetDashboard Cloudwatch: GetInsight RuleReport Cloudwatch: Streamen GetMetric Cloudwatch: ListDashboards Cloudwatch: ListManaged InsightRules Cloudwatch: Streams ListMetric cloudwatch: Detektor PutAnomaly

Servicepräfix	Aktionen
	Cloudwatch: Alarm PutComposite Cloudwatch: PutDashboard cloudwatch: Regel PutInsight Cloudwatch: PutManaged InsightRules Cloudwatch: Alarm PutMetric Cloudwatch: Streamen PutMetric cloudwatch: Bundesstaat SetAlarm cloudwatch: Streams StartMetric cloudwatch: Streams StopMetric

Servicepräfix	Aktionen
codeartifact	Codeartefact: Verbindung AssociateExternal
	codeartifact: Versionen CopyPackage
	Codeartefakt: CreateDomain
	Codeartefakt: CreateRepository
	Codeartefakt: DeleteDomain
	Codeartefakt: DeleteDomain PermissionsPolicy
	Codeartefakt: DeletePackage
	codeartifact: Versionen DeletePackage
	Codeartefakt: DeleteRepository
	Codeartefakt: DeleteRepository PermissionsPolicy
	Codeartefakt: DescribeDomain
	Codeartefakt: DescribePackage
	Codeartefakt: Version DescribePackage
	Codeartefakt: DescribeRepository
	codeartifact: Verbindung DisassociateExternal
	codeartifact: Versionen DisposePackage
	Codeartefakt: GetAssociated PackageGroup
	Codeartefakt: Token GetAuthorization
	Codeartefakt: GetDomain PermissionsPolicy
	Codeartefakt: GetPackage VersionAsset
	Codeartefakt: GetPackage VersionReadme

Servicepräfix	Aktionen
	codeartifact: Endpunkt GetRepository
	Codeartefakt: GetRepository PermissionsPolicy
	Codeartefakt: ListDomains
	codeartifact: Gruppen ListPackage
	Code-Artefakt: ListPackages
	Codeartefakt: ListPackage VersionAssets
	Codeartefakt: ListPackage VersionDependencies
	codeartifact: Versionen ListPackage
	Codeartefakt: ListRepositories
	Codeartefakt: ListRepositories InDomain
	Codeartefakt: Version PublishPackage
	Codeartefakt: PutDomain PermissionsPolicy
	codeartifact: Metadaten PutPackage
	Codeartefakt: PutPackage OriginConfiguration
	Codeartefakt: PutRepository PermissionsPolicy
	codeartifact: Repositorium ReadFrom
	Codeartefakt: UpdatePackage VersionsStatus
	Codeartefakt: UpdateRepository

Servicepräfix	Aktionen
codedeploy	Code bereitstellen: BatchGet ApplicationRevisions codedeploy: Anwendungen BatchGet codedeploy: BatchGet DeploymentGroups CodeDeploy: BatchGet DeploymentInstances codedeploy: Bereitstellungen BatchGet codedeploy: BatchGet DeploymentTargets codedeploy: Instanzen BatchGet OnPremises codedeploy: ContinueDeployment CodeDeploy: CreateApplication CodeDeploy: CreateDeployment codedeploy: Config CreateDeployment codedeploy: Gruppe CreateDeployment codedeploy: DeleteApplication codedeploy: Config DeleteDeployment codedeploy: Gruppe DeleteDeployment codedeploy: Token DeleteGit HubAccount codedeploy: ID DeleteResources ByExternal codedeploy: DeregisterOn PremisesInstance CodeDeploy: GetApplication codedeploy: Überarbeitung GetApplication codedeploy: GetDeployment

Servicepräfix	Aktionen
	codedeploy: Config GetDeployment codedeploy: Gruppe GetDeployment codedeploy: Instanz GetDeployment codedeploy: Ziel GetDeployment codedeploy: GetOn PremisesInstance codedeploy: Überarbeitungen ListApplication codedeploy: ListApplications codedeploy: Konfigurationen ListDeployment codedeploy: Gruppen ListDeployment codedeploy: Instanzen ListDeployment codedeploy: ListDeployments codedeploy: Ziele ListDeployment codedeploy: ListGit HubAccount TokenNames CodeDeploy: ListOn PremisesInstances CodeDeploy: PutLifecycle EventHook ExecutionStatus codedeploy: Überarbeitung RegisterApplication codedeploy: RegisterOn PremisesInstance CodeDeploy: SkipWait TimeFor InstanceTermination CodeDeploy: StopDeployment CodeDeploy: UpdateApplication codedeploy: Gruppe UpdateDeployment

Servicepräfix	Aktionen
codeguru-profiler	codeguru-profiler: Kanäle AddNotification
	codeguru-profiler: BatchGet FrameMetric Daten
	codeguru-profiler: ConfigureAgent
	codeguru-profiler: Gruppe CreateProfiling
	codeguru-profiler: DeleteProfiling Gruppe
	codeguru-profiler: DescribeProfiling Gruppe
	codeguru-profiler: GetFindings ReportAccount Zusammenfassung
	codeguru-profiler: GetNotification Konfiguration
	codeguru-profiler: GetPolicy
	Codeguru-Profiler: GetProfile
	Codeguru-Profiler: GetRecommendations
	codeguru-profiler: Berichte ListFindings
	codeguru-profiler: ListProfile Zeiten
	codeguru-profiler: ListProfiling Gruppen
	codeguru-profiler: PutPermission
	codeguru-profiler: Kanal RemoveNotification
	codeguru-profiler: RemovePermission
	Codeguru-Profiler: SubmitFeedback
	codeguru-profiler: Gruppe UpdateProfiling

Servicepräfix	Aktionen
codeguru-reviewer	Codeguru-Rezensent: AssociateRepository codeguru-reviewer: Rezension CreateCode codeguru-reviewer: Rezension DescribeCode codeguru-reviewer: Feedback DescribeRecommendation codeguru-reviewer: Verband DescribeRepository codeguru-reviewer: DisassociateRepository codeguru-reviewer: Bewertungen ListCode codeguru-reviewer: Feedback ListRecommendation Codeguru-Rezensent: ListRecommendations codeguru-reviewer: Verbände ListRepository codeguru-reviewer: Feedback PutRecommendation

Servicepräfix	Aktionen
codepipeline	Codepipeline: AcknowledgeJob
	Codepipeline: AcknowledgeThird PartyJob
	Codepipeline: CreateCustom ActionType
	Codepipeline: CreatePipeline
	Codepipeline: DeleteCustom ActionType
	Codepipeline: DeletePipeline
	Codepipeline: DeleteWebhook
	codepipeline: Partei DeregisterWebhook WithThird
	codepipeline: Typ GetAction
	codepipeline: Einzelheiten GetJob
	Codepipeline: GetPipeline
	codepipeline: Ausführung GetPipeline
	codepipeline: Bundesstaat GetPipeline
	codepipeline: Einzelheiten GetThird PartyJob
	codepipeline: Ausführungen ListAction
	codepipeline: Typen ListAction
	codepipeline: Ausführungen ListPipeline
	Codepipeline: ListPipelines
	Codepipeline: ListWebhooks
	codepipeline: Stellenangebote PollFor
	codepipeline: Stellenangebote PollFor ThirdParty

Servicepräfix	Aktionen
	codepipeline: Überarbeitung PutAction
	codepipeline: Ergebnis PutApproval
	Codepipeline: PutJob FailureResult
	Codepipeline: PutJob SuccessResult
	Codepipeline: PutThird PartyJob FailureResult
	Codepipeline: PutThird PartyJob SuccessResult
	Codepipeline: PutWebhook
	codepipeline: Partei RegisterWebhook WithThird
	Codepipeline: RollbackStage
	codepipeline: Ausführung StartPipeline
	codepipeline: Ausführung StopPipeline
	codepipeline: Typ UpdateAction
	Codepipeline: UpdatePipeline

Servicepräfix	Aktionen
codestar	codestar: Mitglied AssociateTeam codestar: CreateProject codestar: Profil CreateUser codestar: DeleteProject codestar: Profil DeleteUser codestar: DescribeProject codestar: Profil DescribeUser codestar: Mitglied DisassociateTeam codestar: ListProjects Codestar: ListResources codestar: Mitglieder ListTeam codestar: Profile ListUser Codestar: UpdateProject codestar: Mitglied UpdateTeam codestar: Profil UpdateUser

Servicepräfix	Aktionen
codestar-notifications	codestar-notifications: Regel CreateNotification
	codestar-notifications: Regel DeleteNotification
	Codestar-Benachrichtigungen: DeleteTarget
	codestar-notifications: Regel DescribeNotification
	Codestar-Benachrichtigungen: Typen ListEvent
	codestar-notifications: Regeln ListNotification
	Codestar-Benachrichtigungen: ListTargets
	codestar-notifications:Subscribe
	codestar-notifications:Unsubscribe
	codestar-notifications: Regel UpdateNotification

Servicepräfix	Aktionen
cognito-identity	Cognito-Identity: Pool CreateIdentity kognitive Identität: DeleteIdentities kognitive Identität: Pool DeleteIdentity kognitive Identität: DescribeIdentity kognitive Identität: Pool DescribeIdentity kognitive Identität: GetIdentity PoolRoles kognitive Identität: ListIdentities kognitive Identität: Pools ListIdentity kognitive Identität: Identität LookupDeveloper kognitive Identität: Identitäten MergeDeveloper kognitive Identität: SetIdentity PoolRoles kognitive Identität: Identität UnlinkDeveloper kognitive Identität: Pool UpdateIdentity

Servicepräfix	Aktionen
cognito-idp	cognito-idp: Attribute AddCustom
	cognito-idp: AdminAdd UserTo Gruppe
	cognito-idp: AdminConfirm SignUp
	cognito-idp: Benutzer AdminCreate
	cognito-idp: AdminDelete Benutzer
	cognito-idp: AdminDelete UserAttributes
	cognito-idp: Benutzer AdminDisable ProviderFor
	cognito-idp: AdminDisable Benutzer
	cognito-idp: AdminEnable Benutzer
	cognito-idp: AdminForget Gerät
	cognito-idp: AdminGet Gerät
	cognito-idp: AdminGet Benutzer
	cognito-idp: AdminInitiate Auth
	cognito-idp: AdminLink ProviderFor Benutzer
	cognito-idp: AdminList Geräte
	cognito-idp: AdminList GroupsFor Benutzer
	cognito-idp: AdminList UserAuth Ereignisse
	cognito-idp: AdminRemove UserFrom Gruppe
	cognito-idp: AdminReset UserPassword
	cognito-idp: Herausforderung AdminRespond ToAuth
	cognito-idp: AdminSet Benutzer-MFAP-Referenz

Servicepräfix	Aktionen
	Cognito-IDP: AdminSet UserPassword
	Kognito-IDP: AdminSet UserSettings
	cognito-idp: Rückmeldung AdminUpdate AuthEvent
	Cognito-IDP: AdminUpdate DeviceStatus
	Kognito-IDP: AdminUpdate UserAttributes
	cognito-idp: AdminUser GlobalSign Raus
	cognito-idp: AssociateSoftware Token
	kognito-idp: ChangePassword
	Kognito-IDP: ConfirmDevice
	cognito-idp: Passwort ConfirmForgot
	cognito-idp: ConfirmSign Hoch
	cognito-idp: CreateGroup
	cognito-idp: Anbieter CreateIdentity
	cognito-idp: CreateResource Server
	kognito-idp: CreateUser ImportJob
	cognito-idp: Schwimmbad CreateUser
	kognito-idp: CreateUser PoolClient
	Kognito-IDP: CreateUser PoolDomain
	Kognito-IDP: DeleteGroup
	cognito-idp: Anbieter DeleteIdentity
	cognito-idp: DeleteResource Server

Servicepräfix	Aktionen
	kognito-idp: DeleteUser
	cognito-idp: Attribute DeleteUser
	cognito-idp: DeleteUser Pool
	kognito-idp: DeleteUser PoolClient
	Kognito-IDP: DeleteUser PoolDomain
	cognito-idp: Anbieter DescribeIdentity
	cognito-idp: DescribeResource Server
	cognito-idp: DescribeRisk Konfiguration
	cognito-idp: DescribeUser ImportJob
	cognito-idp: Schwimmbad DescribeUser
	kognito-idp: DescribeUser PoolClient
	Kognito-IDP: DescribeUser PoolDomain
	Kognito-IDP: ForgetDevice
	Kognito-IDP: ForgotPassword
	cognito-idp:GetCSVHeader
	Kognito-IDP: GetDevice
	Kognito-IDP: GetGroup
	cognito-idp: GetIdentity ProviderBy Bezeichner
	cognito-idp: GetLog DeliveryConfiguration
	cognito-idp: Zertifikat GetSigning
	cognito-idp:GetUICustomization

Servicepräfix	Aktionen
	cognito-idp: GetUser
	cognito-idp: Kode GetUser AttributeVerification
	cognito-idp: Config GetUser PoolMfa
	cognito-idp: GlobalSign Raus
	cognito-idp: InitiateAuth
	Kognito-IDP: ListDevices
	Kognito-IDP: ListGroups
	cognito-idp: Anbieter ListIdentity
	cognito-idp: ListResource Server
	cognito-idp: ListUser ImportJobs
	Kognito-IDP: ListUser PoolClients
	cognito-idp: Schwimmbäder ListUser
	Cognito-IDP: ListUsers
	Kognito-IDP: ListUsers InGroup
	cognito-idp: Kode ResendConfirmation
	kognito-idp: RespondTo AuthChallenge
	Kognito-IDP: RevokeToken
	Kognito-IDP: SetLog DeliveryConfiguration
	cognito-idp: Konfiguration SetRisk
	cognito-idp:SetUICustomization
	cognito-idp: SetUser MFAP-Referenz

Servicepräfix	Aktionen
	cognito-idp: Config SetUser PoolMfa
	cognito-idp: SetUser Einstellungen
	cognito-idp: SignUp
	Kognito-IDP: StartUser ImportJob
	Kognito-IDP: StopUser ImportJob
	Kognito-IDP: UpdateAuth EventFeedback
	cognito-idp: Status UpdateDevice
	kognito-idp: UpdateGroup
	cognito-idp: Anbieter UpdateIdentity
	cognito-idp: UpdateResource Server
	cognito-idp: UpdateUser Attribute
	cognito-idp: UpdateUser Pool
	kognito-idp: UpdateUser PoolClient
	Kognito-IDP: UpdateUser PoolDomain
	cognito-idp: Token VerifySoftware
	cognito-idp: VerifyUser Attribut

Servicepräfix	Aktionen
cognito-sync	Cognito-Sync: BulkPublish Kognito-Synchronisierung: DeleteDataset Kognito-Synchronisierung: DescribeDataset Kognito-Synchronisierung: DescribeIdentity PoolUsage cognito-sync: Verwendung DescribeIdentity Cognito-Sync: GetBulk PublishDetails cognito-sync: Ereignisse GetCognito Cognito-Sync: GetIdentity PoolConfiguration Kognito-Synchronisierung: ListDatasets Kognito-Synchronisierung: ListIdentity PoolUsage Kognito-Synchronisierung: ListRecords Kognito-Synchronisierung: RegisterDevice cognito-sync: Ereignisse SetCognito Cognito-Sync: SetIdentity PoolConfiguration cognito-sync: Datensatz SubscribeTo cognito-sync: Datensatz UnsubscribeFrom Cognito-Sync: UpdateRecords

Servicepräfix	Aktionen
comprehendmedical	medizinisch verstehen: Detection V2Job DescribeEntities Comprehend Medical: Beschreiben Sie ICD10cm InferenceJob Comprehend Medical: DescribePhi DetectionJob comparhendmedical: Job DescribeRx NormInference Comprehend Medical: beschreibt NomedCT InferenceJob medizinisch verstehen: V2 DetectEntities comprehendmedical:DetectPHI comprehendmedical:InferICD10CM umfassend medizinisch: Norm InferRx comprehendmedical:InferSNOMEDCT Comprehendmedical: Erkennung von V2-Jobs ListEntities Comprehend Medical: Listic D 10 cm InferenceJobs Comprehend Medical: ListPhi DetectionJobs Comprehendmedical: Stellenangebote ListRx NormInference ComprehendMedical: listet NomedCT auf InferenceJobs Comprehendmedical: Detection V2 Job StartEntities ComprehendMedical: Startic D 10 cm InferenceJob Comprehend Medical: StartPhi DetectionJob comparhendmedical: Job StartRx NormInference ComprehendMedical: Startet NomedCT InferenceJob Comprehendmedical: DetectionV2Job StopEntities

Servicepräfix	Aktionen
	Comprehend Medical: Topic D 10 cm InferenceJob Comprehend Medical: Stopphi DetectionJob comparhendmedical: Job StopRx NormInference ComprehendMedical: Stoppt NomedCT InferenceJob

Servicepräfix	Aktionen
compute-optimizer	compute-optimizer: Einstellungen DeleteRecommendation compute-optimizer: DescribeRecommendation ExportJobs compute-optimizer: Empfehlungen ExportAuto ScalingGroup Compute-Optimizer: Exportiert BS VolumeRecommendations Compute-Optimizer: ExportE2 InstanceRecommendations Compute-Optimizer: Exportiert ECS ServiceRecommendations Computeroptimierer: ExportLambda FunctionRecommendations compute-optimizer: Empfehlungen ExportLicense compute-optimizerRecommendationProjected: GETEC2-Metriken Computeroptimierer: GETECS ServiceRecommendation Projected Metrics Computeroptimierer: GetEffective RecommendationPreferences compute-optimizer: Status GetEnrollment compute-optimizer: Organisation GetEnrollment StatusesFor compute-optimizer: Einstellungen GetRecommendation compute-optimizer: Zusammenfassungen GetRecommendation compute-optimizerPutRecommendation: Einstellungen compute-optimizer: Status UpdateEnrollment

Servicepräfix	Aktionen
config	Konfiguration: BatchGet ResourceConfig
	config: DeleteAggregation Autorisierung
	config: DeleteConfig Regel
	Konfiguration: DeleteConfiguration Aggregator
	Konfiguration: Rekorder DeleteConfiguration
	Konfiguration: DeleteConformance Paket
	config: DeleteDelivery Kanal
	config: DeleteEvaluation Ergebnisse
	Konfiguration: DeleteOrganization ConfigRule
	Konfiguration: DeleteOrganization ConformancePack
	Konfiguration: DeletePending AggregationRequest
	config: DeleteRemediation Konfiguration
	config: DeleteRemediation Ausnahmen
	Config: DeleteResource Konfiguration
	config: DeleteRetention Konfiguration
	config: DeleteStored Abfrage
	config: DeliverConfig Schnappschuss
	Konfiguration: DescribeAggregate ComplianceBy ConfigRules
	Konfiguration: DescribeAggregate ComplianceBy Conforman cePacks
	config: DescribeAggregation Autorisierungen

Servicepräfix	Aktionen
	config: Regel DescribeCompliance ByConfig
	Konfiguration: DescribeCompliance ByResource
	Konfiguration: DescribeConfig RuleEvaluation Status
	config: DescribeConfig Regeln
	config: DescribeConfiguration Aggregatoren
	Konfiguration: Status DescribeConfiguration AggregatorSources
	config: DescribeConfiguration Rekorder
	Konfiguration: DescribeConfiguration RecorderStatus
	Konfiguration: DescribeConformance PackCompliance
	Konfiguration: DescribeConformance Pakete
	Konfiguration: DescribeConformance PackStatus
	config: DescribeDelivery Kanäle
	Konfiguration: DescribeDelivery ChannelStatus
	Konfiguration: DescribeOrganization ConfigRules
	config: DescribeOrganization ConfigRule Status
	Konfiguration: DescribeOrganization ConformancePacks
	config: DescribeOrganization ConformancePack Status
	Konfiguration: DescribePending AggregationRequests
	config: DescribeRemediation Konfigurationen
	config: DescribeRemediation Ausnahmen
	Konfiguration: DescribeRemediation ExecutionStatus

Servicepräfix	Aktionen
	config: DescribeRetention Konfigurationen
	Konfiguration: GetCompliance DetailsBy ConfigRule
	config: GetCompliance DetailsBy Ressource
	Konfiguration: GetCompliance SummaryBy ConfigRule
	Konfiguration: GetCompliance SummaryBy ResourceType
	Konfiguration: GetConformance PackCompliance Einzelheiten
	config: GetConformance PackCompliance Zusammenfassung
	Konfiguration: GetCustom RulePolicy
	Konfiguration: GetDiscovered ResourceCounts
	Konfiguration: GetOrganization ConfigRule DetailedStatus
	Konfiguration: GetOrganization ConformancePack DetailedStatus
	config: GetOrganization CustomRule Richtlinie
	Konfiguration: GetResource ConfigHistory
	Konfiguration: GetResource EvaluationSummary
	config: GetStored Abfrage
	config: ListConformance PackCompliance Ergebnisse
	config: ListDiscovered Ressourcen
	config: ListResource Bewertungen
	config: ListStored Abfragen
	config: PutConfig Regel
	Konfiguration: PutConfiguration Aggregator

Servicepräfix	Aktionen
	Konfiguration: Rekorder PutConfiguration
	Konfiguration: PutConformance Paket
	config: PutDelivery Kanal
	Konfiguration: PutEvaluations
	config: PutExternal Bewertung
	Konfiguration: PutOrganization ConfigRule
	Konfiguration: PutOrganization ConformancePack
	config: PutRemediation Konfigurationen
	config: PutRemediation Ausnahmen
	Config: PutResource Konfiguration
	config: PutRetention Konfiguration
	config: PutStored Abfrage
	Config: SelectResource Konfiguration
	Konfiguration: StartConfig RulesEvaluation
	Konfiguration: StartConfiguration Rekorder
	config: StartRemediation Ausführung
	config: StartResource Bewertung
	Konfiguration: StopConfiguration Rekorder

Servicepräfix	Aktionen
connect	verbinden: ActivateEvaluation Formular verbinden: AssociateApproved Origin verbinden: AssociateBot verbinden: AssociateDefault Wortschatz verbinden: AssociateFlow verbinden: AssociateInstance StorageConfig verbinden: AssociateLambda Funktion verbinden: AssociateLex Bot verbinden: AssociatePhone NumberContact Flow verbinden: AssociateQueue QuickConnects verbinden: AssociateRouting ProfileQueues verbinden: AssociateSecurity Schlüssel verbinden: AssociateUser Fähigkeiten verbinden: BatchGet FlowAssociation verbinden: BatchPut Kontakt verbinden: ClaimPhone Nummer verbinden: CreateAgent Status verbinden: CreateContact Flow verbinden: CreateContact FlowModule verbinden: CreateEvaluation Formular verbinden: CreateHours OfOperation

Servicepräfix	Aktionen
	verbinden: CreateInstance verbinden: CreateIntegration Verband verbinden: CreateParticipant verbinden: CreatePersistent ContactAssociation verbinden: CreatePredefined Attribut verbinden: CreatePrompt verbinden: CreateQueue Connect: CreateQuick Verbinden verbinden: CreateRouting Profil verbinden: CreateRule verbinden: CreateSecurity Profil verbinden: CreateTask Vorlage verbinden: CreateTraffic DistributionGroup verbinden: CreateUse Gehäuse verbinden: CreateUser verbinden: CreateUser HierarchyGroup verbinden: CreateView verbinden: CreateView Version verbinden: CreateVocabulary verbinden: DeactivateEvaluation Formular verbinden: DeleteContact Bewertung

Servicepräfix	Aktionen
	verbinden: DeleteContact Flow verbinden: DeleteContact FlowModule verbinden: DeleteEvaluation Formular verbinden: DeleteHours OfOperation verbinden: DeleteInstance verbinden: DeleteIntegration Verband verbinden: DeletePredefined Attribut verbinden: DeletePrompt verbinden: DeleteQueue Connect: DeleteQuick Verbinden verbinden: DeleteRouting Profil verbinden: DeleteRule verbinden: DeleteSecurity Profil verbinden: DeleteTask Vorlage verbinden: DeleteTraffic DistributionGroup verbinden: DeleteUse Gehäuse verbinden: DeleteUser verbinden: DeleteUser HierarchyGroup verbinden: DeleteView verbinden: DeleteVocabulary verbinden: DescribeAgent Status

Servicepräfix	Aktionen
	verbinden: DescribeContact Bewertung verbinden: DescribeContact Flow verbinden: DescribeContact FlowModule verbinden: DescribeEvaluation Formular verbinden: DescribeInstance Attribut verbinden: DescribeInstance StorageConfig verbinden: DescribePhone Nummer verbinden: DescribeRule verbinden: DescribeTraffic DistributionGroup verbinden: DescribeUser HierarchyGroup verbinden: DescribeUser HierarchyStructure verbinden: DescribeView verbinden: DescribeVocabulary verbinden: DisassociateApproved Origin verbinden: DisassociateBot verbinden: DisassociateFlow verbinden: DisassociateInstance StorageConfig verbinden: DisassociateLambda Funktion verbinden: DisassociateLex Bot verbinden: DisassociatePhone NumberContact Flow verbinden: DisassociateQueue QuickConnects

Servicepräfix	Aktionen
	verbinden: DisassociateRouting ProfileQueues verbinden: DisassociateSecurity Schlüssel verbinden: DisassociateUser Fähigkeiten connect: Kontakt DismissUser verbinden: GetContact Attribute verbinden: GetCurrent MetricData verbinden: GetCurrent UserData verbinden: GetFederation Token verbinden: GetFlow Assoziation verbinden: GetMetric Daten verbinden: GetMetric DataV2 verbinden: Datei GetPrompt verbinden: GetTask Vorlage connect: GetTraffic Vertrieb verbinden: ImportPhone Nummer verbinden: ListApproved Origins verbinden: ListBots connect: ListContact Evaluationen verbinden: ListContact FlowModules verbinden: ListContact Flüsse connect: ListContact Referenzen

Servicepräfix	Aktionen
	connect: ListDefault Vokabeln verbinden: Formulare ListEvaluation verbinden: ListEvaluation FormVersions verbinden: ListFlow Verbände verbinden: ListHours OfOperations verbinden: ListInstance Attribute verbinden: ListInstance StorageConfigs verbinden: ListIntegration Verbände connect: ListLambda Funktionen verbinden: ListLex Bots verbinden: ListPhone Zahlen verbinden: ListPhone NumbersV2 verbinden: Attribute ListPredefined verbinden: ListPrompts verbinden: ListQueue QuickConnects verbinden: ListQueues verbinden: ListQuick Verbindet verbinden: ListRealtime ContactAnalysis SegmentsV2 verbinden: ListRouting ProfileQueues verbinden: ListRouting Profile verbinden: ListRules

Servicepräfix	Aktionen
	verbinden: ListSecurity Schlüssel verbinden: ListSecurity ProfileApplications verbinden: ListSecurity ProfilePermissions verbinden: ListSecurity Profile verbinden: ListTask Vorlagen verbinden: ListTraffic DistributionGroups verbinden: ListUse Fälle verbinden: ListUser HierarchyGroups verbinden: ListUser Fähigkeiten verbinden: ListUsers verbinden: ListViews verbinden: ListView Versionen verbinden: MonitorContact verbinden: PauseContact verbinden: PutUser Status verbinden: ReleasePhone Nummer verbinden: ReplicateInstance verbinden: ResumeContact verbinden: ResumeContact Aufnahme verbinden: SearchAvailable PhoneNumbers verbinden: SearchContacts

Servicepräfix	Aktionen
	verbinden: SearchHours OfOperations verbinden: SearchPredefined Attribute verbinden: SearchPrompts verbinden: SearchQueues verbinden: SearchQuick Verbindet verbinden: SearchRouting Profile verbinden: SearchSecurity Profile verbinden: SearchVocabularies verbinden: SendChat IntegrationEvent verbinden: StartChat Kontakt connect: StartContact Bewertung verbinden: StartContact Aufnahme verbinden: StartContact Streaming verbinden: StartOutbound VoiceContact verbinden: StartTask Kontakt verbinden: StartWeb RTCContact verbinden: StopContact verbinden: StopContact Aufnahme verbinden: StopContact Streaming verbinden: SubmitContact Bewertung verbinden: SuspendContact Aufnahme

Servicepräfix	Aktionen
	verbinden: TransferContact verbinden: UpdateAgent Status verbinden: UpdateContact verbinden: UpdateContact Attribute connect: UpdateContact Bewertung verbinden: UpdateContact FlowContent verbinden: UpdateContact FlowMetadata verbinden: UpdateContact FlowModule Inhalt verbinden: UpdateContact FlowModule Metadaten verbinden: UpdateContact FlowName verbinden: UpdateContact RoutingData verbinden: UpdateContact Zeitplan verbinden: UpdateEvaluation Formular verbinden: UpdateHours OfOperation verbinden: UpdateInstance Attribut verbinden: UpdateInstance StorageConfig verbinden: UpdateParticipant RoleConfig verbinden: UpdatePhone Nummer verbinden: UpdatePhone NumberMetadata verbinden: UpdatePredefined Attribut verbinden: UpdatePrompt

Servicepräfix	Aktionen
	verbinden: UpdateQueue HoursOf Bedienung verbinden: UpdateQueue MaxContacts verbinden: UpdateQueue Name verbinden: UpdateQueue OutboundCaller Config verbinden: UpdateQueue Status verbinden: UpdateQuick ConnectConfig verbinden: UpdateQuick ConnectName verbinden: UpdateRouting ProfileAgent AvailabilityTimer verbinden: UpdateRouting ProfileConcurrency verbinden: UpdateRouting ProfileDefault OutboundQueue verbinden: UpdateRouting ProfileName verbinden: UpdateRouting ProfileQueues verbinden: UpdateRule verbinden: UpdateSecurity Profil verbinden: UpdateTask Vorlage connect: UpdateTraffic Vertrieb verbinden: UpdateUser Hierarchie verbinden: UpdateUser HierarchyGroup Name verbinden: UpdateUser HierarchyStructure verbinden: UpdateUser IdentityInfo verbinden: UpdateUser PhoneConfig

Servicepräfix	Aktionen
	verbinden: UpdateUser Fähigkeiten verbinden: UpdateUser RoutingProfile verbinden: UpdateUser SecurityProfiles verbinden: UpdateView Inhalt verbinden: UpdateView Metadaten
cur	cur: Definition DeleteReport cur: Definitionen DescribeReport cur: Definition ModifyReport cur: Definition PutReport

Servicepräfix	Aktionen
databrew	Datenbraue: BatchDelete RecipeVersion Datenbraue: CreateDataset databrew: Job CreateProfile databrew: CreateProject Datenbraue: CreateRecipe databrew: Job CreateRecipe databrew: CreateRuleset Datenbraue: CreateSchedule Datenbraue: DeleteDataset Datenbraue: DeleteJob Datenbraue: DeleteProject databrew: Version DeleteRecipe Datenbraue: DeleteRuleset Datenbraue: DeleteSchedule Datenbraue: DescribeDataset Datenbraue: DescribeJob databrew: Ausführen DescribeJob databrew: DescribeProject Datenbraue: DescribeRecipe Datenbraue: DescribeRuleset Datenbraue: DescribeSchedule

Servicepräfix	Aktionen
	Datenbraue: ListDatasets databrew: Läuft ListJob databrew: ListJobs Datenbraue: ListProjects Datenbraue: ListRecipes databrew: Versionen ListRecipe databrew: ListRulesets Datenbraue: ListSchedules Datenbraue: PublishRecipe Datenbraue: SendProject SessionAction databrew: Ausführen StartJob databrew: Sitzung StartProject databrew: Ausführen StopJob databrew: UpdateDataset databrew: Job UpdateProfile databrew: UpdateProject Datenbraue: UpdateRecipe databrew: Job UpdateRecipe databrew: UpdateRuleset Datenbraue: UpdateSchedule

Servicepräfix	Aktionen
dataexchange	Datenaustausch: CancelJob
	Datenaustausch: Set CreateData
	Datenaustausch: Aktion CreateEvent
	Datenaustausch: CreateJob
	Datenaustausch: CreateRevision
	Datenaustausch: DeleteAsset
	Datenaustausch: Aktion DeleteEvent
	Datenaustausch: DeleteRevision
	Datenaustausch: Aktion GetEvent
	Datenaustausch: GetJob
	Datenaustausch: ListData SetRevisions
	Datenaustausch: Sätze ListData
	dataexchange: Aktionen ListEvent
	Datenaustausch: ListJobs
	Datenaustausch: Vermögenswerte ListRevision
	Datenaustausch: RevokeRevision
	Datenaustausch: SendData SetNotification
	Datenaustausch: StartJob
	Datenaustausch: UpdateAsset
	Datenaustausch: Set UpdateData
	Datenaustausch: Aktion UpdateEvent

Servicepräfix	Aktionen
	Datenaustausch: UpdateRevision
datapipeline	Datenleitung: ActivatePipeline Datenleitung: CreatePipeline Datenleitung: DeactivatePipeline Datenleitung: DeletePipeline Datenleitung: DescribeObjects Datenleitung: DescribePipelines Datenleitung: EvaluateExpression Datenpipeline: Definition GetPipeline Datenpipeline: ListPipelines datapipeline: Aufgabe PollFor datapipeline: Definition PutPipeline Datenpipeline: QueryObjects datapipeline: Fortschritt ReportTask Datenpipeline: ReportTask RunnerHeartbeat Datenleitung: SetStatus Datenpipeline: Status SetTask Datenpipeline: Definition ValidatePipeline

Servicepräfix	Aktionen
dax	Dax: CreateCluster dax: DecreaseReplication Faktor Dax: DeleteCluster dax: DeleteParameter Gruppe dax: DeleteSubnet Gruppe dax: DescribeClusters dax: DescribeDefault Parameter dax: DescribeEvents dax: DescribeParameter Gruppen dax: DescribeParameters dax: DescribeSubnet Gruppen dax: IncreaseReplication Faktor Dax: RebootNode Dax: UpdateCluster dax: UpdateParameter Gruppe dax: UpdateSubnet Gruppe

Servicepräfix	Aktionen
devicefarm	devicefarm: Pool CreateDevice devicefarm: Profil CreateInstance devicefarm: Profil CreateNetwork devicefarm: CreateProject Gerätefarm: CreateRemote AccessSession Gerätefarm: CreateTest GridProject Gerätefarm: CreateTest GridUrl Gerätefarm: CreateUpload devicefarm:CreateVPCEConfiguration devicefarm: Pool DeleteDevice devicefarm: Profil DeleteInstance devicefarm: Profil DeleteNetwork devicefarm: DeleteProject Gerätefarm: DeleteRemote AccessSession Gerätefarm: DeleteRun Gerätefarm: DeleteTest GridProject Gerätefarm: DeleteUpload devicefarm:DeleteVPCEConfiguration devicefarm: Einstellungen GetAccount devicefarm: GetDevice devicefarm: Instanz GetDevice

Servicepräfix	Aktionen
	devicefarm: Pool GetDevice devicefarm: GetDevice PoolCompatibility devicefarm: Profil GetInstance devicefarm: GetJob devicefarm: Profil GetNetwork devicefarm: Status GetOffering devicefarm: GetProject Gerätefarm: GetRemote AccessSession Gerätefarm: GetRun Gerätefarm: GetSuite Gerätefarm: GetTest Gerätefarm: GetTest GridProject Gerätefarm: GetTest GridSession Gerätefarm: GetUpload devicefarm:GetVPCEConfiguration Gerätefarm: ListArtifacts devicefarm: Instanzen ListDevice devicefarm: Pools ListDevice devicefarm: ListDevices devicefarm: Profile ListInstance devicefarm: ListJobs

Servicepräfix	Aktionen
	devicefarm: Profile ListNetwork devicefarm: Werbeaktionen ListOffering devicefarm: ListOfferings devicefarm: Transaktionen ListOffering devicefarm: ListProjects Gerätefarm: ListRemote AccessSessions Gerätefarm: ListRuns Gerätefarm: ListSamples Gerätefarm: ListSuites Gerätefarm: ListTest GridProjects devicefarm: Aktionen ListTest GridSession devicefarm: Artefakte ListTest GridSession devicefarm: ListTest GridSessions Gerätefarm: ListTests devicefarm: Probleme ListUnique devicefarm: ListUploads devicefarm:ListVPCEConfigurations Gerätefarm: PurchaseOffering Gerätefarm: RenewOffering Gerätefarm: ScheduleRun Gerätefarm: StopJob

Servicepräfix	Aktionen
	Gerätefarm: StopRemote AccessSession Gerätefarm: StopRun devicefarm: Instanz UpdateDevice devicefarm: Pool UpdateDevice devicefarm: Profil UpdateInstance devicefarm: Profil UpdateNetwork devicefarm: UpdateProject Gerätefarm: UpdateTest GridProject Gerätefarm: UpdateUpload devicefarm:UpdateVPCEConfiguration

Servicepräfix	Aktionen
devops-guru	devops-guru: Kanal AddNotification
	Devops-Guru: DeleteInsight
	Devops-Guru: Health DescribeAccount
	devops-guru: Überblick DescribeAccount
	Devops-Guru: DescribeAnomaly
	Devops-Guru: DescribeEvent SourcesConfig
	Devops-Guru: DescribeFeedback
	Devops-Guru: DescribeInsight
	Devops-Guru: Health DescribeOrganization
	devops-guru: Überblick DescribeOrganization
	Devops-Guru: Health DescribeOrganization ResourceCollection
	Devops-Guru: DescribeResource CollectionHealth
	Devops-Guru: Integration DescribeService
	devops-guru: Schätzung GetCost
	devops-guruGetResource: Sammlung
	Devops-Guru: ListAnomalies ForInsight
	Devops-Guru: ListAnomalous LogGroups
	Devops-Guru: ListEvents
	Devops-Guru: ListInsights
	devops-guru: Ressourcen ListMonitored
	devops-guru: Kanäle ListNotification

Servicepräfix	Aktionen
	devops-guru: Einblicke ListOrganization Devops-Guru: ListRecommendations Devops-Guru: PutFeedback devops-guru: Kanal RemoveNotification Devops-Guru: SearchInsights Devops-Guru: Einblicke SearchOrganization devops-guru: Schätzung StartCost Devops-Guru: UpdateEvent SourcesConfig devops-guru: Sammlung UpdateResource devops-guru: Integration UpdateService

Servicepräfix	Aktionen
directconnect	direkte Verbindung: AcceptDirect ConnectGateway AssociationProposal Direktverbindung: AllocateConnection OnInterconnect directconnect: Verbindung AllocateHosted Direktverbindung: AllocatePrivate VirtualInterface Direktverbindung: AllocatePublic VirtualInterface Direktverbindung: AllocateTransit VirtualInterface Direktverbindung: AssociateConnection WithLag directconnect: Verbindung AssociateHosted Direktverbindung: AssociateMac SecKey directconnect: Schnittstelle AssociateVirtual Direktverbindung: ConfirmConnection directconnect: Vereinbarung ConfirmCustomer directconnect: ConfirmPrivate VirtualInterface Direktverbindung: ConfirmPublic VirtualInterface Direktverbindung: ConfirmTransit VirtualInterface directconnect:CreateBGPPeer Direktverbindung: CreateConnection Direktverbindung: CreateDirect ConnectGateway directconnect: Assoziation CreateDirect ConnectGateway Direktverbindung: CreateDirect ConnectGateway AssociationProposal

Servicepräfix	Aktionen
	Direktverbindung: CreateInterconnect Direktverbindung: CreateLag Direktverbindung: CreatePrivate VirtualInterface Direktverbindung: CreatePublic VirtualInterface Direktverbindung: CreateTransit VirtualInterface directconnect:DeleteBGPPeer Direktverbindung: DeleteConnection Direktverbindung: DeleteDirect ConnectGateway directconnect: Assoziation DeleteDirect ConnectGateway Direktverbindung: DeleteDirect ConnectGateway Associati onProposal Direktverbindung: DeleteInterconnect Direktverbindung: DeleteLag directconnect: Schnittstelle DeleteVirtual directconnect: Loa DescribeConnection Direktverbindung: DescribeConnections Direktverbindung: DescribeConnections OnInterconnect directconnect: Metadaten DescribeCustomer Direktverbindung: DescribeDirect ConnectGateway Associati onProposals directconnect: Assoziationen DescribeDirect ConnectGateway directconnect: Anlagen DescribeDirect ConnectGateway

Servicepräfix	Aktionen
	Direktverbindung: DescribeDirect ConnectGateways directconnect: Verbindungen DescribeHosted directconnect: Loa DescribeInterconnect Direktverbindung: DescribeInterconnects Direktverbindung: DescribeLags Direktverbindung: DescribeLoa Direktverbindung: DescribeLocations directconnect: Konfiguration DescribeRouter directconnect: Gateways DescribeVirtual directconnect: Schnittstellen DescribeVirtual Direktverbindung: DisassociateConnection FromLag Direktverbindung: DisassociateMac SecKey directconnect: Geschichte ListVirtual InterfaceTest directconnect: StartBgp FailoverTest Direktverbindung: StopBgp FailoverTest Direktverbindung: UpdateConnection Direktverbindung: UpdateDirect ConnectGateway directconnect: Assoziation UpdateDirect ConnectGateway Direktverbindung: UpdateLag Direktverbindung: UpdateVirtual InterfaceAttributes

Servicepräfix	Aktionen
dlm	dlm: Richtlinie CreateLifecycle
	dlm: Richtlinie DeleteLifecycle
	dlm: Richtlinien GetLifecycle
	dlm: Richtlinie GetLifecycle
	dlm: Richtlinie UpdateLifecycle

Servicepräfix	Aktionen
dms	dms: ApplyPending MaintenanceAction dms: Empfehlungen BatchStart dms: Ausführen CancelReplication TaskAssessment dms: Anbieter CreateData dms: CreateEndpoint dms: Abonnement CreateEvent dms: Profil CreateInstance dms: Projekt CreateMigration dms: Config CreateReplication dms: Instanz CreateReplication dms: CreateReplication SubnetGroup dms: Aufgabe CreateReplication dms: DeleteCertificate dms: DeleteConnection dms: Anbieter DeleteData dms: DeleteEndpoint dms: Abonnement DeleteEvent dms: DeleteFleet AdvisorCollector dms: DeleteFleet AdvisorDatabases dms: Profil DeleteInstance dms: Projekt DeleteMigration

Servicepräfix	Aktionen
	dms: Config DeleteReplication dms: Instanz DeleteReplication dms: DeleteReplication SubnetGroup dms: Aufgabe DeleteReplication dms: Ausführen DeleteReplication TaskAssessment dms: Attribute DescribeAccount dms: DescribeApplicable IndividualAssessments dms: DescribeCertificates dms: DescribeConnections dms: DescribeEndpoints dms: Einstellungen DescribeEndpoint dms: Typen DescribeEndpoint dms: Versionen DescribeEngine dms: Kategorien DescribeEvent dms: DescribeEvents dms: Abonnements DescribeEvent dms: DescribeFleet AdvisorCollectors dms: DescribeFleet AdvisorDatabases dms: Analyse DescribeFleet AdvisorLsa dms: DescribeFleet AdvisorSchema ObjectSummary dms: DescribeFleet AdvisorSchemas

Servicepräfix	Aktionen
	dms: DescribeMetadata ModelImports dms: DescribeOrderable ReplicationInstances dms: DescribePending MaintenanceActions dms: Einschränkungen DescribeRecommendation dms: DescribeRecommendations dms: DescribeRefresh SchemasStatus dms: Konfigurationen DescribeReplication dms: Instanzen DescribeReplication dms: Protokolle DescribeReplication InstanceTask dms: DescribeReplications dms: DescribeReplication SubnetGroups dms: DescribeReplication TableStatistics dms: Ergebnisse DescribeReplication TaskAssessment dms: Läuft DescribeReplication TaskAssessment dms: Bewertungen DescribeReplication TaskIndividual dms: Aufgaben DescribeReplication dms: DescribeSchemas dms: Statistiken DescribeTable dms: ExportMetadata ModelAssessment dms: Modell GetMetadata dms: ImportCertificate

Servicepräfix	Aktionen
	dms: ListMetadata ModelAssessment ActionItems dms: ModifyEndpoint dms: Abonnement ModifyEvent dms: Config ModifyReplication dms: Instanz ModifyReplication dms: ModifyReplication SubnetGroup dms: Aufgabe ModifyReplication dms: Aufgabe MoveReplication dms: Instanz RebootReplication dms: RefreshSchemas dms: Tabellen ReloadReplication dms: ReloadTables dms: Analyse RunFleet AdvisorLsa dms: StartMetadata ModelAssessment dms: StartMetadata ModelConversion dms: StartMetadata ModelExport ToTarget dms: StartRecommendations dms: StartReplication dms: Aufgabe StartReplication dms: StartReplication TaskAssessment dms: Aufgabe StopReplication

Servicepräfix	Aktionen
	dms: TestConnection dms: Brücke UpdateSubscriptions ToEvent
docdb-elastic	docdb-elastic: Schnappschuss CopyCluster docdb-elastisch: DeleteCluster docdb-elastic: Schnappschuss DeleteCluster docdb-elastisch: GetCluster docdb-elastic: Schnappschuss GetCluster docdb-elastisch: ListClusters docdb-elastic: Schnappschüsse ListCluster docdb-elastisch: RestoreCluster FromSnapshot docdb-elastisch: StartCluster docdb-elastisch: StopCluster docdb-elastisch: UpdateCluster

Servicepräfix	Aktionen
ds	ds: Verzeichnis AcceptShared ds: AddIp Routen ds: AddRegion ds: CancelSchema Erweiterung ds: ConnectDirectory ds: CreateAlias ds: CreateComputer ds: CreateConditional Spediteur ds: CreateDirectory ds: CreateLog Abonnement ds: CreateMicrosoft AD Anzeigen: CreateSnapshot ds: CreateTrust ds: DeleteConditional Spediteur ds: DeleteDirectory ds: DeleteLog Abonnement ds: DeleteSnapshot ds: DeleteTrust ds: DeregisterCertificate ds: DeregisterEvent Thema ds: DescribeCertificate

Servicepräfix	Aktionen
	ds: DescribeClient AuthenticationSettings ds: DescribeConditional Spediteure ds: DescribeDirectories ds: DescribeDomain Steuerungen ds: DescribeEvent Themen ds:DescribeLDAPSSettings ds: DescribeRegions ds: DescribeSettings ds: DescribeShared Verzeichnisse ds: DescribeSnapshots ds: DescribeTrusts ds: DescribeUpdate Verzeichnis ds: DisableClient Authentifizierung ds:DisableLDAPS ds: DisableRadius ds: DisableSso ds: EnableClient Authentifizierung ds:EnableLDAPS ds: EnableRadius ds: EnableSso ds: GetDirectory Grenzwerte

Servicepräfix	Aktionen
	ds: GetSnapshot Grenzen ds: ListCertificates ds: ListIp Routen ds: ListLog Abonnements ds: ListSchema Erweiterungen ds: RegisterCertificate ds: RegisterEvent Thema ds: RejectShared Verzeichnis ds: RemoveIp Routen ds: RemoveRegion ds: ResetUser Passwort ds: RestoreFrom Schnappschuss ds: ShareDirectory ds: StartSchema Erweiterung ds: UnshareDirectory ds: UpdateConditional Spediteur ds: Einrichtung UpdateDirectory ds: UpdateNumber OfDomain Steuerungen ds: UpdateRadius ds: UpdateSettings ds: UpdateTrust

Servicepräfix	Aktionen
	ds: VerifyTrust

Servicepräfix	Aktionen
dynamodb	Dynamodb: CreateBackup dynamodb: Tabelle CreateGlobal dynamodb: CreateTable Dynamodb: DeleteBackup Dynamodb: DeleteTable Dynamodb: DescribeBackup dynamodb: Backups DescribeContinuous dynamodb: Einblicke DescribeContributor dynamodb: DescribeEndpoints Dynamodb: DescribeExport dynamodb: Tabelle DescribeGlobal dynamodb: DescribeGlobal TableSettings Dynamodb: DescribeImport Dynamodb: DescribeKinesis StreamingDestination Dynamodb: DescribeLimits Dynamodb: DescribeStream Dynamodb: DescribeTable dynamodb: Skalierung DescribeTable ReplicaAuto Dynamodb: DescribeTime ToLive Dynamodb: DisableKinesis StreamingDestination Dynamodb: EnableKinesis StreamingDestination

Servicepräfix	Aktionen
	Dynamodb: ExportTable ToPoint InTime
	dynamodb: Richtlinie GetResource
	dynamodb: ImportTable
	Dynamodb: ListBackups
	dynamodb: Einblicke ListContributor
	dynamodb: ListExports
	dynamodb: Tabellen ListGlobal
	dynamodb: ListImports
	Dynamodb: ListStreams
	Dynamodb: ListTables
	Dynamodb: RestoreTable FromBackup
	Dynamodb: RestoreTable ToPoint InTime
	dynamodb: Backups UpdateContinuous
	dynamodb: Einblicke UpdateContributor
	dynamodb: Tabelle UpdateGlobal
	dynamodb: UpdateGlobal TableSettings
	Dynamodb: UpdateKinesis StreamingDestination
	Dynamodb: UpdateTable
	dynamodb: Skalierung UpdateTable ReplicaAuto
	Dynamodb: UpdateTime ToLive

Servicepräfix	Aktionen
ebs	ebs: CompleteSnapshot ebs: StartSnapshot

Servicepräfix	Aktionen
ec2	ec2: Übertragung AcceptAddress
	ec2: Zitat AcceptReserved InstancesExchange
	ec2: AcceptTransit GatewayMulticast DomainAssociations
	ec2: Anlage AcceptTransit GatewayPeering
	ec2: Anlage AcceptTransit GatewayVpc
	ec2: AcceptVpc EndpointConnections
	ec2: AcceptVpc PeeringConnection
	ec2: Apfelwein AdvertiseByoip
	ec2: AllocateAddress
	ec2: AllocateHosts
	ec2: AllocateIpam PoolCidr
	ec2: ApplySecurity GroupsTo ClientVpn TargetNetwork
	ec2:6 Adressen AssignIpv
	ec2: AssignPrivate IpAddresses
	ec2: Adresse AssignPrivate NatGateway
	ec2: AssociateAddress
	ec2: Netzwerk AssociateClient VpnTarget
	ec2: Optionen AssociateDhcp
	ec2: Rolle AssociateEnclave CertificateIam
	ec2: AssociateIam InstanceProfile
	ec2: AssociateInstance EventWindow

Servicepräfix	Aktionen
	ec2: Von yosan AssociateIpam
	ec2: AssociateIpam ResourceDiscovery
	ec2: AssociateNat GatewayAddress
	ec2: Tabelle AssociateRoute
	ec2: AssociateSubnet CidrBlock
	ec2: Domäne AssociateTransit GatewayMulticast
	ec2: Tabelle AssociateTransit GatewayPolicy
	ec2: Tabelle AssociateTransit GatewayRoute
	ec2: Schnittstelle AssociateTrunk
	ec2: AssociateVpc CidrBlock
	ec2: AttachClassic LinkVpc
	ec2: Gateway AttachInternet
	ec2: Schnittstelle AttachNetwork
	ec2: Anbieter AttachVerified AccessTrust
	ec2: AttachVolume
	ec2: Gateway AttachVpn
	ec2: AuthorizeClient VpnIngress
	ec2: AuthorizeSecurity GroupEgress
	ec2: AuthorizeSecurity GroupIngress
	ec2: BundleInstance
	ec2: Aufgabe CancelBundle

Servicepräfix	Aktionen
	ec2: Reservierung CancelCapacity
	ec2: CancelCapacity ReservationFleets
	ec2: Aufgabe CancelConversion
	ec2: Aufgabe CancelExport
	ec2: CancelImage LaunchPermission
	ec2: Aufgabe CancelImport
	ec2: CancelReserved InstancesListing
	ec2: CancelSpot FleetRequests
	ec2: CancelSpot InstanceRequests
	ec2: Instanz ConfirmProduct
	ec2: Bild CopyFpga
	ec2: CopyImage
	ec2: CopySnapshot
	ec2: Reservierung CreateCapacity
	ec2: CreateCapacity ReservationFleet
	ec2: Gateway CreateCarrier
	ec2: CreateClient VpnEndpoint
	ec2: CreateClient VpnRoute
	ec2: Apfelwein CreateCoip
	ec2: Schwimmbad CreateCoip
	ec2: Gateway CreateCustomer

Servicepräfix	Aktionen
	ec2: Subnetz CreateDefault
	ec2: Vpc CreateDefault
	ec2: Optionen CreateDhcp
	ec2: Gateway CreateEgress OnlyInternet
	ec2: CreateFleet
	ec2: Protokolle CreateFlow
	ec2: Bild CreateFpga
	ec2: CreateImage
	ec2: CreateInstance ConnectEndpoint
	ec2: CreateInstance EventWindow
	ec2: CreateInstance ExportTask
	ec2: Gateway CreateInternet
	ec2: CreateIpam
	ec2: Schwimmbad CreateIpam
	ec2: CreateIpam ResourceDiscovery
	ec2: Geltungsbereich CreateIpam
	ec2: Paar CreateKey
	ec2: Vorlage CreateLaunch
	ec2: CreateLaunch TemplateVersion
	ec2: CreateLocal GatewayRoute
	ec2: Tabelle CreateLocal GatewayRoute

Servicepräfix	Aktionen
	ec2: Assoziation CreateLocal GatewayRoute TableVirtual Interface Group
	ec2: Verband CreateLocal GatewayRoute TableVpc
	ec2: CreateManaged PrefixList
	ec2: Gateway CreateNat
	ec2: Acl CreateNetwork
	ec2: CreateNetwork AclEntry
	ec2: Umfang CreateNetwork InsightsAccess
	ec2: CreateNetwork InsightsPath
	ec2: Schnittstelle CreateNetwork
	ec2: CreateNetwork InterfacePermission
	ec2: Gruppe CreatePlacement
	ec2: Ipv4Pool CreatePublic
	CreateReplaceRootVolumeec2: Aufgabe
	ec2: CreateReserved InstancesListing
	ec2: CreateRestore ImageTask
	ec2: CreateRoute
	ec2: Tabelle CreateRoute
	ec2: Gruppe CreateSecurity
	ec2: CreateSnapshot
	ec2: CreateSnapshots

Servicepräfix	Aktionen
	ec2: CreateSpot DatafeedSubscription
	ec2: CreateStore ImageTask
	ec2: CreateSubnet
	ec2: CreateSubnet CidrReservation
	ec2: CreateTraffic MirrorFilter
	ec2: Regel CreateTraffic MirrorFilter
	ec2: CreateTraffic MirrorSession
	ec2: CreateTraffic MirrorTarget
	ec2: Gateway CreateTransit
	ec2: CreateTransit GatewayConnect
	ec2: Gleichaltriger CreateTransit GatewayConnect
	ec2: Domäne CreateTransit GatewayMulticast
	ec2: Anlage CreateTransit GatewayPeering
	ec2: Tabelle CreateTransit GatewayPolicy
	ec2: CreateTransit GatewayPrefix ListReference
	ec2: CreateTransit GatewayRoute
	ec2: Tabelle CreateTransit GatewayRoute
	ec2: CreateTransit GatewayRoute TableAnnouncement
	ec2: Anlage CreateTransit GatewayVpc
	ec2: CreateVerified AccessEndpoint
	ec2: CreateVerified AccessGroup

Servicepräfix	Aktionen
	ec2: CreateVerified AccessInstance
	ec2: Anbieter CreateVerified AccessTrust
	ec2: CreateVolume
	ec2: CreateVpc
	ec2: Endpunkt CreateVpc
	ec2: Benachrichtigung CreateVpc EndpointConnection
	ec2: Konfiguration CreateVpc EndpointService
	ec2: CreateVpc PeeringConnection
	ec2: Verbindung CreateVpn
	ec2: CreateVpn ConnectionRoute
	ec2: Gateway CreateVpn
	ec2: Gateway DeleteCarrier
	ec2: DeleteClient VpnEndpoint
	ec2: DeleteClient VpnRoute
	ec2: Apfelwein DeleteCoip
	ec2: Schwimmbad DeleteCoip
	ec2: Gateway DeleteCustomer
	ec2: Optionen DeleteDhcp
	ec2: Gateway DeleteEgress OnlyInternet
	ec2: DeleteFleets
	ec2: Protokolle DeleteFlow

Servicepräfix	Aktionen
	ec2: Bild DeleteFpga
	ec2: DeleteInstance ConnectEndpoint
	ec2: DeleteInstance EventWindow
	ec2: Gateway DeleteInternet
	ec2: DeleteIpam
	ec2: Schwimmbad DeleteIpam
	ec2: DeleteIpam ResourceDiscovery
	ec2: Geltungsbereich DeleteIpam
	ec2: Paar DeleteKey
	ec2: Vorlage DeleteLaunch
	ec2: DeleteLaunch TemplateVersions
	ec2: DeleteLocal GatewayRoute
	ec2: Tabelle DeleteLocal GatewayRoute
	ec2: Assoziation DeleteLocal GatewayRoute TableVirtual Interface Group
	ec2: Verband DeleteLocal GatewayRoute TableVpc
	ec2: DeleteManaged PrefixList
	ec2: Gateway DeleteNat
	ec2: Acl DeleteNetwork
	ec2: DeleteNetwork AclEntry
	ec2: Umfang DeleteNetwork InsightsAccess

Servicepräfix	Aktionen
	ec2: DeleteNetwork InsightsAccess ScopeAnalysis
	ec2: DeleteNetwork InsightsAnalysis
	ec2: DeleteNetwork InsightsPath
	ec2: Schnittstelle DeleteNetwork
	ec2: DeleteNetwork InterfacePermission
	ec2: Gruppe DeletePlacement
	ec2: Ipv4Pool DeletePublic
	ec2: DeleteQueued ReservedInstances
	ec2: DeleteRoute
	ec2: Tabelle DeleteRoute
	ec2: Gruppe DeleteSecurity
	ec2: DeleteSnapshot
	ec2: DeleteSpot DatafeedSubscription
	ec2: DeleteSubnet
	ec2: DeleteSubnet CidrReservation
	ec2: DeleteTraffic MirrorFilter
	ec2: Regel DeleteTraffic MirrorFilter
	ec2: DeleteTraffic MirrorSession
	ec2: DeleteTraffic MirrorTarget
	ec2: Gateway DeleteTransit
	ec2: DeleteTransit GatewayConnect

Servicepräfix	Aktionen
	ec2: Gleichaltriger DeleteTransit GatewayConnect
	ec2: Domäne DeleteTransit GatewayMulticast
	ec2: Anlage DeleteTransit GatewayPeering
	ec2: Tabelle DeleteTransit GatewayPolicy
	ec2: DeleteTransit GatewayPrefix ListReference
	ec2: DeleteTransit GatewayRoute
	ec2: Tabelle DeleteTransit GatewayRoute
	ec2: DeleteTransit GatewayRoute TableAnnouncement
	ec2: Anlage DeleteTransit GatewayVpc
	ec2: DeleteVerified AccessEndpoint
	ec2: DeleteVerified AccessGroup
	ec2: DeleteVerified AccessInstance
	ec2: Anbieter DeleteVerified AccessTrust
	ec2: DeleteVolume
	ec2: DeleteVpc
	ec2: Benachrichtigungen DeleteVpc EndpointConnection
	ec2: Endpunkte DeleteVpc
	ec2: Konfigurationen DeleteVpc EndpointService
	ec2: DeleteVpc PeeringConnection
	ec2: Verbindung DeleteVpn
	ec2: DeleteVpn ConnectionRoute

Servicepräfix	Aktionen
	ec2: Gateway DeleteVpn
	ec2: Apfelwein DeprovisionByoip
	ec2: Von yosan DeprovisionIpam
	ec2: DeprovisionIpam PoolCidr
	ec2: IPv4 DeprovisionPublic PoolCidr
	ec2: DeregisterImage
	ec2: Attribute DeregisterInstance EventNotification
	ec2: DeregisterTransit GatewayMulticast GroupMembers
	ec2: DeregisterTransit GatewayMulticast GroupSources
	ec2: Attribute DescribeAccount
	ec2: DescribeAddresses
	ec2: Attribut DescribeAddresses
	ec2: Übertragungen DescribeAddress
	ec2: DescribeAggregate IdFormat
	ec2: Zonen DescribeAvailability
	ec2: DescribeAws NetworkPerformance MetricSubscriptions
	ec2: Aufgaben DescribeBundle
	ec2: Apfelweine DescribeByoip
	ec2: DescribeCapacity ReservationFleets
	ec2: Reservierungen DescribeCapacity
	ec2: Gateways DescribeCarrier

Servicepräfix	Aktionen
	ec2: DescribeClassic LinkInstances
	ec2: Regeln DescribeClient VpnAuthorization
	ec2: DescribeClient VpnConnections
	ec2: DescribeClient VpnEndpoints
	ec2: DescribeClient VpnRoutes
	ec2: Netzwerke DescribeClient VpnTarget
	ec2: Schwimmbäder DescribeCoip
	ec2: Aufgaben DescribeConversion
	ec2: Gateways DescribeCustomer
	ec2: Optionen DescribeDhcp
	ec2: Gateways DescribeEgress OnlyInternet
	ec2: GPUs DescribeElastic
	ec2: DescribeExport ImageTasks
	ec2: Aufgaben DescribeExport
	ec2: DescribeFast LaunchImages
	ec2: DescribeFast SnapshotRestores
	ec2: Geschichte DescribeFleet
	ec2: Instanzen DescribeFleet
	ec2: DescribeFleets
	ec2: Protokolle DescribeFlow
	ec2: DescribeFpga ImageAttribute

Servicepräfix	Aktionen
	ec2: Bilder DescribeFpga
	ec2: DescribeHost ReservationOfferings
	ec2: Reservierungen DescribeHost
	ec2: DescribeHosts
	ec2: Verbände DescribeIam InstanceProfile
	ec2: DescribeIdentity IdFormat
	ec2: Format DescribeId
	ec2: Attribut DescribeImage
	ec2: DescribeImages
	ec2: DescribeImport ImageTasks
	ec2: DescribeImport SnapshotTasks
	ec2: Attribut DescribeInstance
	ec2: DescribeInstance ConnectEndpoints
	ec2: DescribeInstance CreditSpecifications
	ec2: Attribute DescribeInstance EventNotification
	ec2: DescribeInstance EventWindows
	ec2: DescribeInstances
	ec2: Status DescribeInstance
	ec2: Topologie DescribeInstance
	ec2: DescribeInstance TypeOfferings
	ec2: Typen DescribeInstance

Servicepräfix	Aktionen
	ec2: Gateways DescribeInternet
	ec2: Byosan DescribeIpam
	ec2DescribeIpam: Schwimmbecken
	ec2: DescribeIpam ResourceDiscoveries
	ec2: Verbände DescribeIpam ResourceDiscovery
	ec2: DescribeIpams
	ec2: Bereiche DescribeIpam
	ec2:6 Pools DescribeIpv
	ec2: Paare DescribeKey
	ec2: Vorlagen DescribeLaunch
	ec2: DescribeLaunch TemplateVersions
	ec2: Tabellen DescribeLocal GatewayRoute
	ec2: Verbände DescribeLocal GatewayRoute TableVirtual Interface Group
	ec2: Verbände DescribeLocal GatewayRoute TableVpc
	ec2: Gateways DescribeLocal
	ec2: DescribeLocal GatewayVirtual InterfaceGroups
	ec2: Schnittstellen DescribeLocal GatewayVirtual
	ec2: Schnappschüsse DescribeLocked
	ec2: Gastgeber DescribeMac
	ec2: DescribeManaged PrefixLists

Servicepräfix	Aktionen
	ec2: Adressen DescribeMoving
	ec2: Gateways DescribeNat
	ec2: Acls DescribeNetwork
	ec2: DescribeNetwork InsightsAccess ScopeAnalyses
	ec2: Geltungsbereiche DescribeNetwork InsightsAccess
	ec2: DescribeNetwork InsightsAnalyses
	ec2: DescribeNetwork InsightsPaths
	ec2: DescribeNetwork InterfaceAttribute
	ec2: DescribeNetwork InterfacePermissions
	ec2: Schnittstellen DescribeNetwork
	ec2: Gruppen DescribePlacement
	ec2: Listen DescribePrefix
	ec2: DescribePrincipal IdFormat
	ec2: IPv4-Pools DescribePublic
	ec2: DescribeRegions
	ec2: Aufgaben DescribeReplace RootVolume
	ec2: Instanzen DescribeReserved
	ec2: DescribeReserved InstancesListings
	ec2: DescribeReserved InstancesModifications
	ec2: DescribeReserved InstancesOfferings
	ec2: Tabellen DescribeRoute

Servicepräfix	Aktionen
	ec2: DescribeScheduled InstanceAvailability
	ec2: Instanzen DescribeScheduled
	ec2: DescribeSecurity GroupReferences
	ec2: DescribeSecurity GroupRules
	ec2: Gruppen DescribeSecurity
	ec2: Attribut DescribeSnapshot
	ec2: DescribeSnapshots
	ec2: DescribeSnapshot TierStatus
	ec2: DescribeSpot DatafeedSubscription
	ec2: DescribeSpot FleetInstances
	ec2: Geschichte DescribeSpot FleetRequest
	ec2: DescribeSpot FleetRequests
	ec2: DescribeSpot InstanceRequests
	ec2: DescribeSpot PriceHistory
	ec2: DescribeStale SecurityGroups
	ec2: DescribeStore ImageTasks
	ec2: DescribeSubnets
	ec2: DescribeTraffic MirrorFilters
	ec2: DescribeTraffic MirrorSessions
	ec2: DescribeTraffic MirrorTargets
	ec2: DescribeTransit GatewayAttachments

Servicepräfix	Aktionen
	ec2: Gleichaltrige DescribeTransit GatewayConnect
	ec2: DescribeTransit GatewayConnects
	ec2: Domänen DescribeTransit GatewayMulticast
	ec2: Anlagen DescribeTransit GatewayPeering
	ec2: Tabellen DescribeTransit GatewayPolicy
	ec2: DescribeTransit GatewayRoute TableAnnouncements
	ec2: Tabellen DescribeTransit GatewayRoute
	ec2: Gateways DescribeTransit
	ec2: Anlagen DescribeTransit GatewayVpc
	ec2: DescribeTrunk InterfaceAssociations
	ec2: DescribeVerified AccessEndpoints
	ec2: DescribeVerified AccessGroups
	ec2: DescribeVerified AccessInstance LoggingConfigurations
	ec2: DescribeVerified AccessInstances
	ec2: Anbieter DescribeVerified AccessTrust
	ec2: Attribut DescribeVolume
	ec2: DescribeVolumes
	ec2: Änderungen DescribeVolumes
	ec2: Status DescribeVolume
	ec2: Attribut DescribeVpc
	ec2: DescribeVpc ClassicLink

Servicepräfix	Aktionen
	ec2: DescribeVpc ClassicLink DnsSupport
	ec2: Benachrichtigungen DescribeVpc EndpointConnection
	ec2: DescribeVpc EndpointConnections
	ec2: Endpunkte DescribeVpc
	ec2: Konfigurationen DescribeVpc EndpointService
	ec2: Berechtigungen DescribeVpc EndpointService
	ec2: DescribeVpc EndpointServices
	ec2: DescribeVpc PeeringConnections
	ec2: DescribeVpcs
	ec2: Verbindungen DescribeVpn
	ec2: Gateways DescribeVpn
	ec2: DetachClassic LinkVpc
	ec2: Gateway DetachInternet
	ec2: Schnittstelle DetachNetwork
	ec2: Anbieter DetachVerified AccessTrust
	ec2: DetachVolume
	ec2: Gateway DetachVpn
	ec2: Übertragung DisableAddress
	ec2: DisableAws NetworkPerformance MetricSubscription
	ec2: Standard DisableEbs EncryptionBy
	ec2: Starten DisableFast

Servicepräfix	Aktionen
	ec2: DisableFast SnapshotRestores
	ec2: DisableImage
	ec2: Zugriff DisableImage BlockPublic
	ec2: Veraltet DisableImage
	ec2: DisableImage DeregistrationProtection
	ec2: Konto DisableIpam OrganizationAdmin
	ec2: DisableSerial ConsoleAccess
	ec2: Zugriff DisableSnapshot BlockPublic
	ec2: DisableTransit GatewayRoute TablePropagation
	ec2: DisableVgw RoutePropagation
	ec2: DisableVpc ClassicLink
	ec2: DisableVpc ClassicLink DnsSupport
	ec2: DisassociateAddress
	ec2: Netzwerk DisassociateClient VpnTarget
	ec2: Rolle DisassociateEnclave CertificateIam
	ec2: DisassociateIam InstanceProfile
	ec2: DisassociateInstance EventWindow
	ec2: Von yosan DisassociateIpam
	ec2: DisassociateIpam ResourceDiscovery
	ec2: DisassociateNat GatewayAddress
	ec2: Tabelle DisassociateRoute

Servicepräfix	Aktionen
	ec2: DisassociateSubnet CidrBlock
	ec2: Domäne DisassociateTransit GatewayMulticast
	ec2: Tabelle DisassociateTransit GatewayPolicy
	ec2: Tabelle DisassociateTransit GatewayRoute
	ec2: Schnittstelle DisassociateTrunk
	ec2: DisassociateVpc CidrBlock
	ec2: Übertragung EnableAddress
	ec2: EnableAws NetworkPerformance MetricSubscription
	ec2: Standard EnableEbs EncryptionBy
	ec2: Starten EnableFast
	ec2: EnableFast SnapshotRestores
	ec2: EnableImage
	ec2: Zugriff EnableImage BlockPublic
	ec2: Veraltet EnableImage
	ec2: EnableImage DeregistrationProtection
	ec2: Konto EnableIpam OrganizationAdmin
	ec2: Teilen EnableReachability AnalyzerOrganization
	ec2: EnableSerial ConsoleAccess
	ec2: Zugriff EnableSnapshot BlockPublic
	ec2: EnableTransit GatewayRoute TablePropagation
	ec2: EnableVgw RoutePropagation

Servicepräfix	Aktionen
	ec2: IO EnableVolume
	ec2: EnableVpc ClassicLink
	ec2: EnableVpc ClassicLink DnsSupport
	ec2: Liste ExportClient VpnClient CertificateRevocation
	ec2: Konfiguration ExportClient VpnClient
	ec2: ExportImage
	ec2: ExportTransit GatewayRoutes
	ec2: GetAssociated EnclaveCertificate IamRoles
	ec2: IPv6 GetAssociated PoolCidrs
	ec2: Daten GetAws NetworkPerformance
	ec2: GetCapacity ReservationUsage
	ec2: GetCoip PoolUsage
	ec2: Ausgabe GetConsole
	ec2: Bildschirmfoto GetConsole
	ec2: GetDefault CreditSpecification
	ec2: GetEbs DefaultKms KeyId
	ec2: Standard GetEbs EncryptionBy
	ec2: Vorlage GetFlow LogsIntegration
	ec2: Reservierung GetGroups ForCapacity
	ec2: Vorschau GetHost ReservationPurchase
	ec2: GetImage BlockPublic AccessState

Servicepräfix	Aktionen
	ec2: GetInstance MetadataDefaults
	ec2: Kneipe GetInstance TpmEk
	ec2: GetInstance TypesFrom InstanceRequirements
	ec2: GetInstance UefiData
	ec2: GetIpam AddressHistory
	ec2: GetIpam DiscoveredAccounts
	ec2: Adressen GetIpam DiscoveredPublic
	ec2: Apfelweine GetIpam DiscoveredResource
	ec2: GetIpam PoolAllocations
	ec2: GetIpam PoolCidrs
	ec2: GetIpam ResourceCidrs
	ec2: GetLaunch TemplateData
	ec2: Verbände GetManaged PrefixList
	ec2: Einträge GetManaged PrefixList
	ec2: Ergebnisse GetNetwork InsightsAccess ScopeAnalysis
	ec2: GetNetwork InsightsAccess ScopeContent
	ec2: Daten GetPassword
	ec2: Zitat GetReserved InstancesExchange
	ec2: Vpc GetSecurity GroupsFor
	ec2: Status GetSerial ConsoleAccess
	ec2: GetSnapshot BlockPublic AccessState

Servicepräfix	Aktionen
	ec2: GetSpot PlacementScores
	ec2: GetSubnet CidrReservations
	ec2: Propagationen GetTransit GatewayAttachment
	ec2: GetTransit GatewayMulticast DomainAssociations
	ec2: GetTransit GatewayPolicy TableAssociations
	ec2: GetTransit GatewayPolicy TableEntries
	ec2: GetTransit GatewayPrefix ListReferences
	ec2: GetTransit GatewayRoute TableAssociations
	ec2: GetTransit GatewayRoute TablePropagations
	ec2: Richtlinie GetVerified AccessEndpoint
	ec2: Richtlinie GetVerified AccessGroup
	ec2: GetVpn ConnectionDevice SampleConfiguration
	ec2: Typen GetVpn ConnectionDevice
	ec2: Status GetVpn TunnelReplacement
	ec2: Liste ImportClient VpnClient CertificateRevocation
	ec2: ImportImage
	ec2: ImportInstance
	ec2: Paar ImportKey
	ec2: ImportSnapshot
	ec2: ImportVolume
	ec2: Mülleimer ListImages InRecycle

Servicepräfix	Aktionen
	ec2: Mülleimer ListSnapshots InRecycle
	ec2: LockSnapshot
	ec2: Attribut ModifyAddress
	ec2: ModifyAvailability ZoneGroup
	ec2: Reservierung ModifyCapacity
	ec2: ModifyCapacity ReservationFleet
	ec2: ModifyClient VpnEndpoint
	ec2: ModifyDefault CreditSpecification
	ec2: ModifyEbs DefaultKms KeyId
	ec2: ModifyFleet
	ec2: ModifyFpga ImageAttribute
	ec2: ModifyHosts
	ec2: ModifyIdentity IdFormat
	ec2: Format ModifyId
	ec2: Attribut ModifyImage
	ec2: Attribut ModifyInstance
	ec2: Attribute ModifyInstance CapacityReservation
	ec2: ModifyInstance CreditSpecification
	ec2: Zeit ModifyInstance EventStart
	ec2: ModifyInstance EventWindow
	ec2: ModifyInstance MaintenanceOptions

Servicepräfix	Aktionen
	ec2: ModifyInstance MetadataDefaults
	ec2: ModifyInstance MetadataOptions
	ec2: Platzierung ModifyInstance
	ec2: ModifyIpam
	ec2: Schwimmbad ModifyIpam
	ec2: ModifyIpam ResourceCidr
	ec2: ModifyIpam ResourceDiscovery
	ec2: Geltungsbereich ModifyIpam
	ec2: Vorlage ModifyLaunch
	ec2: ModifyLocal GatewayRoute
	ec2: ModifyManaged PrefixList
	ec2: ModifyNetwork InterfaceAttribute
	ec2: Optionen ModifyPrivate DnsName
	ec2: Instanzen ModifyReserved
	ec2: ModifySecurity GroupRules
	ec2: Attribut ModifySnapshot
	ec2: Stufe ModifySnapshot
	ec2: ModifySpot FleetRequest
	ec2: Attribut ModifySubnet
	ec2: ModifyTraffic MirrorFilter NetworkServices
	ec2: Regel ModifyTraffic MirrorFilter

Servicepräfix	Aktionen
	ec2: ModifyTraffic MirrorSession
	ec2: Gateway ModifyTransit
	ec2: ModifyTransit GatewayPrefix ListReference
	ec2: Anlage ModifyTransit GatewayVpc
	ec2: ModifyVerified AccessEndpoint
	ec2: Richtlinie ModifyVerified AccessEndpoint
	ec2: ModifyVerified AccessGroup
	ec2: Richtlinie ModifyVerified AccessGroup
	ec2: ModifyVerified AccessInstance
	ec2: ModifyVerified AccessInstance LoggingConfiguration
	ec2: Anbieter ModifyVerified AccessTrust
	ec2: ModifyVolume
	ec2: Attribut ModifyVolume
	ec2: Attribut ModifyVpc
	ec2: Endpunkt ModifyVpc
	ec2: Benachrichtigung ModifyVpc EndpointConnection
	ec2: Konfiguration ModifyVpc EndpointService
	ec2: ModifyVpc EndpointService PayerResponsibility
	ec2: Berechtigungen ModifyVpc EndpointService
	ec2: Optionen ModifyVpc PeeringConnection
	ec2: Mietverhältnis ModifyVpc

Servicepräfix	Aktionen
	ec2: Verbindung ModifyVpn
	ec2: ModifyVpn ConnectionOptions
	ec2: ModifyVpn TunnelCertificate
	ec2: ModifyVpn TunnelOptions
	ec2: MonitorInstances
	ec2: MoveAddress ToVpc
	ec2: IPAM MoveByoip CidrTo
	ec2: Apfelwein ProvisionByoip
	ec2: Von yosan ProvisionIpam
	ec2: ProvisionIpam PoolCidr
	ec2: IPv4 ProvisionPublic PoolCidr
	ec2: Reservierung PurchaseHost
	ec2: PurchaseReserved InstancesOffering
	ec2: Instanzen PurchaseScheduled
	ec2: RebootInstances
	ec2: RegisterImage
	ec2: Attribute RegisterInstance EventNotification
	ec2: RegisterTransit GatewayMulticast GroupMembers
	ec2: RegisterTransit GatewayMulticast GroupSources
	ec2: RejectTransit GatewayMulticast DomainAssociations
	ec2: Anlage RejectTransit GatewayPeering

Servicepräfix	Aktionen
	ec2: Anlage RejectTransit GatewayVpc
	ec2: RejectVpc EndpointConnections
	ec2: RejectVpc PeeringConnection
	ec2: ReleaseAddress
	ec2: ReleaseHosts
	ec2: ReleaseIpam PoolAllocation
	ec2: Verband ReplaceIam InstanceProfile
	ec2: ReplaceNetwork AclAssociation
	ec2: ReplaceNetwork AclEntry
	ec2: ReplaceRoute
	ec2: ReplaceRoute TableAssociation
	ec2: ReplaceTransit GatewayRoute
	ec2: Tunnel ReplaceVpn
	ec2: Status ReportInstance
	ec2: Flotte RequestSpot
	ec2: Instanzen RequestSpot
	ec2: Attribut ResetAddress
	ec2: ResetEbs DefaultKms KeyId
	ec2: ResetFpga ImageAttribute
	ec2: Attribut ResetImage
	ec2: Attribut ResetInstance

Servicepräfix	Aktionen
	ec2: ResetNetwork InterfaceAttribute
	ec2: Attribut ResetSnapshot
	ec2: RestoreAddress ToClassic
	ec2: Mülleimer RestoreImage FromRecycle
	ec2: Ausführung RestoreManaged PrefixList
	ec2: Behälter RestoreSnapshot FromRecycle
	ec2: Stufe RestoreSnapshot
	ec2: RevokeClient VpnIngress
	ec2: RevokeSecurity GroupEgress
	ec2: RevokeSecurity GroupIngress
	ec2: RunInstances
	ec2: Instanzen RunScheduled
	ec2: SearchLocal GatewayRoutes
	ec2: Gruppen SearchTransit GatewayMulticast
	ec2: SearchTransit GatewayRoutes
	ec2: Unterbrechen SendDiagnostic
	ec2: StartInstances
	ec2: StartNetwork InsightsAccess ScopeAnalysis
	ec2: StartNetwork InsightsAnalysis
	ec2: Überprüfung StartVpc EndpointService PrivateDns
	ec2: StopInstances

Servicepräfix	Aktionen
	ec2: TerminateClient VpnConnections
	ec2: TerminateInstances
	ec2:6 Adressen UnassignIpv
	ec2: UnassignPrivate IpAddresses
	ec2: Adresse UnassignPrivate NatGateway
	ec2: UnlockSnapshot
	ec2: UnmonitorInstances
	ec2: UpdateSecurity GroupRule DescriptionsEgress
	ec2: UpdateSecurity GroupRule DescriptionsIngress
	ec2: Apfelwein WithdrawByoip

Servicepräfix	Aktionen
ecr	ecr: BatchCheck LayerAvailability ecr: Bild BatchDelete ecr: Bild BatchGet ecr: Konfiguration BatchGet RepositoryScanning ecr: Hochladen CompleteLayer ecr: Regel CreatePull ThroughCache ecr: CreateRepository ecr: CreateRepository CreationTemplate ecr: Richtlinie DeleteLifecycle ecr: Regel DeletePull ThroughCache ecr: Richtlinie DeleteRegistry ecr: DeleteRepository ecr: DeleteRepository CreationTemplate ecr: Richtlinie DeleteRepository ecr: DescribeImage ReplicationStatus ecr: DescribeImages ecr: DescribeImage ScanFindings ecr: Regeln DescribePull ThroughCache ecr: DescribeRegistry ecr: DescribeRepositories ecr: Wertmarke GetAuthorization

Servicepräfix	Aktionen
	ecr: Ebene GetDownload UrlFor ecr: Richtlinie GetLifecycle ecr: GetLifecycle PolicyPreview ecr: Richtlinie GetRegistry ecr: GetRegistry ScanningConfiguration ecr: Richtlinie GetRepository ecr: Hochladen InitiateLayer ecr: ListImages ecr: PutImage ecr: PutImage ScanningConfiguration ecr: Richtlinie PutRegistry ecr: PutRegistry ScanningConfiguration ecr: Konfiguration PutReplication ecr: Scannen StartImage ecr: StartLifecycle PolicyPreview ecr: Regel UpdatePull ThroughCache ecr: Teil UploadLayer ecr: Regel ValidatePull ThroughCache

Servicepräfix	Aktionen
ecr-public	ecr-public: BatchCheck LayerAvailability
	ecr-public: Bild BatchDelete
	ecr-public: Hochladen CompleteLayer
	ecr-public: CreateRepository
	ecr-public: DeleteRepository
	ecr-public: Richtlinie DeleteRepository
	ecr-public: DescribeImages
	ecr-public: DescribeRegistries
	ecr-public: DescribeRepositories
	ecr-public: Token GetAuthorization
	ecr-public: GetRegistry CatalogData
	ecr-public: GetRepository CatalogData
	ecr-public: Richtlinie GetRepository
	ecr-public: Hochladen InitiateLayer
	ecr-public: PutImage
	ecr-public: PutRegistry CatalogData
	ecr-public: PutRepository CatalogData
	ecr-public: Richtlinie SetRepository
	ecr-public: Teil UploadLayer

Servicepräfix	Aktionen
ecs	ecs: Anbieter CreateCapacity ecs: CreateCluster ecs: CreateService ecs: CreateTask Satz ecs: DeleteAccount Einstellung ecs: DeleteAttributes ecs: DeleteCapacity Anbieter ecs: DeleteCluster ecs: DeleteService ecs: DeleteTask Definitionen ecs: DeleteTask Satz ecs: DeregisterContainer Instanz ecs: DeregisterTask Definition ecs: DescribeCapacity Anbieter ecs: DescribeClusters ecs: DescribeContainer Instanzen ecs: DescribeServices ecs: DescribeTask Definition ecs: DescribeTasks ecs: DescribeTask Sets ecs: DiscoverPoll Endpunkt

Servicepräfix	Aktionen
	ecs: ExecuteCommand ecs: GetTask Schutz ecs: ListAccount Einstellungen ecs: ListAttributes ecs: ListClusters ecs: ListContainer Instanzen ecs: ListServices ecs: ListServices ByNamespace ecs: ListTask DefinitionFamilies ecs: ListTask Definitionen ecs: ListTasks ecs: PutAccount Einstellung ecs: PutAccount SettingDefault ecs: PutAttributes ecs: PutCluster CapacityProviders ecs: RegisterContainer Instanz ecs: RegisterTask Definition ecs: RunTask ecs: StartTask ecs: StopTask ecs: SubmitAttachment StateChanges

Servicepräfix	Aktionen
	ecs: SubmitContainer StateChange ecs: SubmitTask StateChange ecs: UpdateCapacity Anbieter ecs: UpdateCluster ecs: UpdateCluster Einstellungen ecs: UpdateContainer Agent ecs: UpdateContainer InstancesState ecs: UpdateService ecs: UpdateService PrimaryTask Satz ecs: UpdateTask Schutz ecs: UpdateTask Set

Servicepräfix	Aktionen
eks	eks: AssociateAccess Richtlinie
	eks: AssociateEncryption Config
	eks: AssociateIdentity ProviderConfig
	eks: CreateAccess Eintrag
	eks: CreateAddon
	eks: CreateCluster
	eks: CreateEks AnywhereSubscription
	eks: CreateFargate Profil
	eks: CreateNodegroup
	eks: DeleteAccess Eintrag
	eks: DeleteAddon
	eks: DeleteCluster
	eks: DeleteEks AnywhereSubscription
	eks: DeleteFargate Profil
	eks: DeleteNodegroup
	eks: DeletePod IdentityAssociation
	eks: DeregisterCluster
	eks: DescribeAccess Eintrag
	eks: DescribeAddon
	eks: DescribeAddon Konfiguration
	eks: DescribeAddon Versionen

Servicepräfix	Aktionen
	eks: DescribeCluster
	eks: DescribeEks AnywhereSubscription
	eks: DescribeFargate Profil
	eks: DescribeIdentity ProviderConfig
	eks: DescribeInsight
	eks: DescribeNodegroup
	eks: DescribePod IdentityAssociation
	eks: DescribeUpdate
	eks: DisassociateAccess Politik
	eks: DisassociateIdentity ProviderConfig
	eks: ListAccess Einträge
	eks: ListAccess Richtlinien
	eks: ListAddons
	eks: ListAssociated AccessPolicies
	eks: ListClusters
	eks: ListEks AnywhereSubscriptions
	eks: ListFargate Profile
	eks: ListIdentity ProviderConfigs
	eks: ListInsights
	eks: ListNodegroups
	eks: ListPod IdentityAssociations

Servicepräfix	Aktionen
	eks: ListUpdates eks: RegisterCluster eks: UpdateAccess Eintrag eks: UpdateAddon eks: UpdateCluster Config eks: UpdateCluster Version eks: UpdateEks AnywhereSubscription eks: UpdateNodegroup Config eks: UpdateNodegroup Version eks: UpdatePod IdentityAssociation
elastic-inference	Elastic-Inference: Angebote DescribeAccelerator elastische Inferenz: DescribeAccelerators elastische Inferenz: Typen DescribeAccelerator

Servicepräfix	Aktionen
elasticache	elasticache: Eintritt AuthorizeCache SecurityGroup
	elastischer Schmerz: BatchApply UpdateAction
	elastischer Schmerz: BatchStop UpdateAction
	elastischer Schmerz: CompleteMigration
	elastischer Schmerz: CopyServerless CacheSnapshot
	elastischer Schmerz: CopySnapshot
	Elasticache: Cluster CreateCache
	elastischer Cache: CreateCache ParameterGroup
	elastischer Schmerz: CreateCache SecurityGroup
	elastischer Schmerz: CreateCache SubnetGroup
	elastischer Schmerz: CreateGlobal ReplicationGroup
	elasticache: Gruppe CreateReplication
	elasticache: Cache CreateServerless
	elastischer Cache: CreateServerless CacheSnapshot
	elastischer Schmerz: CreateSnapshot
	elastischer Schmerz: CreateUser
	elasticache: Gruppe CreateUser
	elasticache: Gruppe DecreaseNode GroupsIn GlobalReplication
	elasticache: Anzahl DecreaseReplica
	elasticache: Cluster DeleteCache
	elastischer Cache: DeleteCache ParameterGroup

Servicepräfix	Aktionen
	elastischer Schmerz: DeleteCache SecurityGroup
	elastischer Schmerz: DeleteCache SubnetGroup
	elastischer Schmerz: DeleteGlobal ReplicationGroup
	elasticache: Gruppe DeleteReplication
	elasticache: Cache DeleteServerless
	elastischer Cache: DeleteServerless CacheSnapshot
	elastischer Schmerz: DeleteSnapshot
	elastischer Schmerz: DeleteUser
	elasticache: Gruppe DeleteUser
	elasticache: Cluster DescribeCache
	elastischer Cache: DescribeCache EngineVersions
	elastischer Schmerz: DescribeCache ParameterGroups
	elasticache: Parameter DescribeCache
	Elasticache: DescribeCache SecurityGroups
	elastischer Schmerz: DescribeCache SubnetGroups
	elastischer Schmerz: DescribeEngine DefaultParameters
	elastischer Schmerz: DescribeEvents
	elastischer Schmerz: DescribeGlobal ReplicationGroups
	elasticache: Gruppen DescribeReplication
	Elasticache: DescribeReserved CacheNodes
	elasticache: Angebote DescribeReserved CacheNodes

Servicepräfix	Aktionen
	elasticache: Caches DescribeServerless
	elastischer Cache: DescribeServerless CacheSnapshots
	elasticache: Aktualisierungen DescribeService
	Elasticache: DescribeSnapshots
	elasticache: Aktionen DescribeUpdate
	elasticache: Gruppen DescribeUser
	Elasticache: DescribeUsers
	elastischer Schmerz: DisassociateGlobal ReplicationGroup
	elastischer Schmerz: ExportServerless CacheSnapshot
	elastischer Schmerz: FailoverGlobal ReplicationGroup
	elasticache: Gruppe IncreaseNode GroupsIn GlobalReplication
	elasticache: Anzahl IncreaseReplica
	elasticache: Änderungen ListAllowed NodeType
	elasticache: Cluster ModifyCache
	elastischer Cache: ModifyCache ParameterGroup
	elastischer Schmerz: ModifyCache SubnetGroup
	elastischer Schmerz: ModifyGlobal ReplicationGroup
	elasticache: Gruppe ModifyReplication
	elasticache: Konfiguration ModifyReplication GroupShard
	elasticache: Cache ModifyServerless
	elastischer Cache: ModifyUser

Servicepräfix	Aktionen
	elasticache: Gruppe ModifyUser
	elasticache: Angebot PurchaseReserved CacheNodes
	Elasticache: RebalanceSlots InGlobal ReplicationGroup
	Elasticache: Cluster RebootCache
	elastischer Cache: ResetCache ParameterGroup
	Elasticache: Eintritt RevokeCache SecurityGroup
	elastischer Schmerz: StartMigration
	elastischer Schmerz: TestFailover
	elastischer Schmerz: TestMigration

Servicepräfix	Aktionen
elasticbeanstalk	elasticbeanstalk: Aktualisieren AbortEnvironment Elasticbeanstalk: ApplyEnvironment ManagedAction elastische Bohnenstange: AssociateEnvironment OperationsRole elasticbeanstalk:CheckDNSAvailability elastische Bohnenstange: ComposeEnvironments elastische Bohnenstange: CreateApplication elasticbeanstalk: Ausführung CreateApplication elasticbeanstalk: Vorlage CreateConfiguration elasticbeanstalk: CreateEnvironment elasticbeanstalk: Ausführung CreatePlatform elasticbeanstalk: Standort CreateStorage elasticbeanstalk: DeleteApplication elasticbeanstalk: Ausführung DeleteApplication elasticbeanstalk: Vorlage DeleteConfiguration elasticbeanstalk: Konfiguration DeleteEnvironment elasticbeanstalk: Version DeletePlatform elasticbeanstalk: Attribute DescribeAccount elasticbeanstalk: DescribeApplications elasticbeanstalk: Versionen DescribeApplication elasticbeanstalk: Optionen DescribeConfiguration elasticbeanstalk: Einstellungen DescribeConfiguration

Servicepräfix	Aktionen
	elasticbeanstalk: Health DescribeEnvironment elasticbeanstalk: Geschichte DescribeEnvironment ManagedAction elasticbeanstalk: DescribeEnvironment ManagedActions elasticbeanstalk: Ressourcen DescribeEnvironment elasticbeanstalk: DescribeEnvironments elastische Bohnenstange: DescribeEvents elasticbeanstalk: Health DescribeInstances elasticbeanstalk: Ausführung DescribePlatform elastische Beanstalk: DisassociateEnvironment OperationsRole elastische Bohnenstange: ListAvailable SolutionStacks elasticbeanstalk: Zweige ListPlatform elasticbeanstalk: Versionen ListPlatform elasticbeanstalk: RebuildEnvironment elasticbeanstalk: Informationen RequestEnvironment elasticbeanstalk: Server RestartApp elasticbeanstalk: Informationen RetrieveEnvironment elasticbeanstalk: CNAMEs SwapEnvironment ElasticBeanstalk: TerminateEnvironment elastische Bohnenstange: UpdateApplication elastische Bohnenstange: UpdateApplication ResourceLifecycle elasticbeanstalk: Ausführung UpdateApplication

Servicepräfix	Aktionen
	elasticbeanstalk: Vorlage UpdateConfiguration elasticbeanstalk: UpdateEnvironment elasticbeanstalk: Einstellungen ValidateConfiguration

Servicepräfix	Aktionen
elasticfilesystem	elastisches Dateisystem: Punkt CreateAccess
	elastisches Dateisystem: System CreateFile
	elastisches Dateisystem: Ziel CreateMount
	elastisches Dateisystem: Konfiguration CreateReplication
	elastisches Dateisystem: Punkt DeleteAccess
	elastisches Dateisystem: System DeleteFile
	elastisches Dateisystem: DeleteFile SystemPolicy
	elastisches Dateisystem: Ziel DeleteMount
	elastisches Dateisystem: Konfiguration DeleteReplication
	elastisches Dateisystem: Punkte DescribeAccess
	elasticfilesystem: Einstellungen DescribeAccount
	elasticfilesystem: Richtlinie DescribeBackup
	elastisches Dateisystem: DescribeFile SystemPolicy
	elastisches Dateisystem: Systeme DescribeFile
	elastisches Dateisystem: Konfiguration DescribeLifecycle
	elastisches Dateisystem: Ziele DescribeMount
	elastisches Dateisystem: Gruppen DescribeMount TargetSecurity
	elastisches Dateisystem: Konfigurationen DescribeReplication
	elastisches Dateisystem: Gruppen ModifyMount TargetSecurity
	elasticfilesystem: Einstellungen PutAccount
	elasticfilesystem: Richtlinie PutBackup

Servicepräfix	Aktionen
	elastisches Dateisystem: PutFile SystemPolicy elastisches Dateisystem: Konfiguration PutLifecycle elastisches Dateisystem: System UpdateFile elastisches Dateisystem: UpdateFile SystemProtection

Servicepräfix	Aktionen
elasticloadbalancing	elasticloadbalancing: Zertifikate AddListener elastischer Lastenausgleich: AddTrust StoreRevocations elastischer Lastenausgleich: ApplySecurity GroupsTo LoadBalancer elasticloadbalancing: Subnetze AttachLoad BalancerTo elasticloadbalancing: Prüfen ConfigureHealth elasticloadbalancing: Richtlinie CreateApp CookieStickiness Elastic Load Balancing: ELB-Richtlinie erstellen CookieStickiness elastischer Lastenausgleich: CreateListener elasticloadbalancing: Balancer CreateLoad elastischer Lastenausgleich: CreateLoad BalancerListeners elastischer Lastenausgleich: CreateLoad BalancerPolicy elastischer Lastenausgleich: CreateRule elasticloadbalancing: Gruppe CreateTarget elasticloadbalancing: Speichern CreateTrust elastischer Lastenausgleich: DeleteListener elasticloadbalancing: Balancer DeleteLoad elastischer Lastenausgleich: DeleteLoad BalancerListeners elastischer Lastenausgleich: DeleteLoad BalancerPolicy elastischer Lastenausgleich: DeleteRule elasticloadbalancing: Gruppe DeleteTarget elasticloadbalancing: Speichern DeleteTrust

Servicepräfix	Aktionen
	elasticloadbalancing: Balancer DeregisterInstances FromLoad elastischer Lastenausgleich: DeregisterTargets elasticloadbalancing: Grenzwerte DescribeAccount elasticloadbalancing: Health DescribeInstance elasticloadbalancing: Zertifikate DescribeListener elastischer Lastenausgleich: DescribeListeners elastischer Lastenausgleich: DescribeLoad BalancerAttributes elastischer Lastenausgleich: DescribeLoad BalancerPolicies elasticloadbalancing: Typen DescribeLoad BalancerPolicy elasticloadbalancing: Balancer DescribeLoad elastischer Lastenausgleich: DescribeRules elasticloadbalancing:DescribeSSLPolicies elastischer Lastenausgleich: DescribeTarget GroupAttributes elasticloadbalancing: Gruppen DescribeTarget elasticloadbalancing: Health DescribeTarget elastischer Lastenausgleich: DescribeTrust StoreAssociations elastischer Lastenausgleich: DescribeTrust StoreRevocations elasticloadbalancing: Speichert DescribeTrust elasticloadbalancing: Subnetze DetachLoad BalancerFrom elastischer Lastenausgleich: DisableAvailability ZonesFor LoadBalancer

Servicepräfix	Aktionen
	elastischer Lastenausgleich: EnableAvailability ZonesFor LoadBalancer elastischer Lastenausgleich: GetTrust StoreCa CertificatesBundle elasticloadbalancing: Inhalt GetTrust StoreRevocation elastischer Lastenausgleich: ModifyListener elastischer Lastenausgleich: ModifyLoad BalancerAttributes elastischer Lastenausgleich: ModifyRule elasticloadbalancing: Gruppe ModifyTarget elastischer Lastenausgleich: ModifyTarget GroupAttributes elasticloadbalancing: Speichern ModifyTrust elasticloadbalancing: Balancer RegisterInstances WithLoad elastischer Lastenausgleich: RegisterTargets elasticloadbalancing: Zertifikate RemoveListener elastischer Lastenausgleich: RemoveTrust StoreRevocations elastischer Lastenausgleich: SetIp AddressType elasticloadbalancing: SSL-Zertifikat SetLoad BalancerListener elasticloadbalancing SetLoad BalancerPoliciesForBackend: Server elastischer Lastenausgleich: SetLoad BalancerPolicies OfListener elasticloadbalancing: Prioritäten SetRule elasticloadbalancing: Gruppen SetSecurity elastischer Lastenausgleich: SetSubnets

Servicepräfix	Aktionen
elastictranscoder	elastischer Transcoder: CancelJob
	elastischer Transcoder: CreateJob
	elastischer Transcoder: CreatePipeline
	elastischer Transcoder: CreatePreset
	elastischer Transcoder: DeletePipeline
	elastischer Transcoder: DeletePreset
	elastischer Transcoder: ListJobs ByPipeline
	elastischer Transcoder: ListJobs ByStatus
	elastischer Transcoder: ListPipelines
	elastischer Transcoder: ListPresets
	elastischer Transcoder: ReadJob
	elastischer Transcoder: ReadPipeline
	elastischer Transcoder: ReadPreset
	elastischer Transcoder: TestRole
	elastischer Transcoder: UpdatePipeline
	elastictranscoder: Benachrichtigungen UpdatePipeline
	elastictranscoder: Status UpdatePipeline

Servicepräfix	Aktionen
emr-containers	emr-containers: Ausführen CancelJob emr-containers: Vorlage CreateJob emr-containers: Endpunkt CreateManaged emr-containers: Konfiguration CreateSecurity emr-containers: Cluster CreateVirtual emr-containers: Vorlage DeleteJob emr-containers: Endpunkt DeleteManaged emr-containers: Cluster DeleteVirtual emr-containers: Ausführen DescribeJob emr-containers: Vorlage DescribeJob emr-containers: Endpunkt DescribeManaged emr-containers: Konfiguration DescribeSecurity emr-containers: Cluster DescribeVirtual emr-containers: Anmeldeinformationen GetManaged EndpointSession emr-containers: Läuft ListJob emr-containers: Vorlagen ListJob emr-containers: Endpunkte ListManaged emr-containers: Konfigurationen ListSecurity emr-containers: Cluster ListVirtual emr-containers: Ausführen StartJob

Servicepräfix	Aktionen
emr-serverless	emr-serverless: Ausführen CancelJob
	emr-serverless: CreateApplication
	emr-serverlos: DeleteApplication
	emr-serverlos: GetApplication
	emr-serverless: Ausführen GetDashboard ForJob
	emr-serverless: Ausführen GetJob
	emr-serverless: ListApplications
	emr-serverless: Läuft ListJob
	emr-serverlos: StartApplication
	emr-serverless: Ausführen StartJob
	emr-serverless: StopApplication
	emr-serverlos: UpdateApplication

Servicepräfix	Aktionen
es	es: Verbindung AcceptInbound ja: AcceptInbound CrossCluster SearchConnection ja: AssociatePackage ja: AuthorizeVpc EndpointAccess ja: CancelElasticsearch ServiceSoftware Aktualisieren ja: CancelService SoftwareUpdate ja: CreateDomain es: CreateElasticsearch Domäne es: CreateOutbound Verbindung ja: CreateOutbound CrossCluster SearchConnection ja: CreatePackage es: CreateVpc Endpunkt ja: DeleteDomain es: DeleteElasticsearch Domäne ja: DeleteElasticsearch ServiceRole es: DeleteInbound Verbindung ja: DeleteInbound CrossCluster SearchConnection es: DeleteOutbound Verbindung ja: DeleteOutbound CrossCluster SearchConnection ja: DeletePackage es: DeleteVpc Endpunkt

Servicepräfix	Aktionen
	ja: DescribeDomain ja: DescribeDomain AutoTunes ja: DescribeDomain ChangeProgress Ja: DescribeDomain Config es: DescribeDomain Health es: DescribeDomain Knoten ja: DescribeDomains ja: DescribeDry RunProgress es: DescribeElasticsearch Domäne ja: DescribeElasticsearch DomainConfig es: DescribeElasticsearch Domänen es: DescribeElasticsearch InstanceType Grenzwerte es: DescribeInbound Verbindungen ja: DescribeInbound CrossCluster SearchConnections ja: DescribeInstance TypeLimits es: DescribeOutbound Verbindungen ja: DescribeOutbound CrossCluster SearchConnections ja: DescribePackages es: DescribeReserved ElasticsearchInstance Angebote ja: DescribeReserved ElasticsearchInstances ja: DescribeReserved InstanceOfferings

Servicepräfix	Aktionen
	es: DescribeReserved Instanzen es: DescribeVpc Endpunkte ja: DissociatePackage ja: GetCompatible ElasticsearchVersions es: GetCompatible Versionen es: GetData Quelle ja: GetDomain MaintenanceStatus ja: GetPackage VersionHistory es: GetUpgrade Geschichte es: GetUpgrade Status es: ListData Quellen es: ListDomain Namen ja: ListDomains ForPackage ja: ListElasticsearch InstanceTypes es: ListElasticsearch Versionen ja: ListInstance TypeDetails ja: ListPackages ForDomain es: ListScheduled Aktionen ja: ListVersions ja: ListVpc EndpointAccess es: ListVpc Endpunkte

Servicepräfix	Aktionen
	es: Domäne ListVpc EndpointsFor es: PurchaseReserved ElasticsearchInstance Angebot ja: PurchaseReserved InstanceOffering es: RejectInbound Verbindung ja: RejectInbound CrossCluster SearchConnection ja: RevokeVpc EndpointAccess es: StartDomain Wartung es: StartElasticsearch ServiceSoftware Aktualisierung ja: StartService SoftwareUpdate es: UpdateData Quelle Ja: UpdateDomain Config ja: UpdateElasticsearch DomainConfig ja: UpdatePackage ja: UpdateScheduled Aktion es: UpdateVpc Endpunkt ja: UpgradeDomain es: UpgradeElasticsearch Domäne

Servicepräfix	Aktionen
Veranstaltungen	Ereignisse: ActivateEvent Quelle
	Ereignisse: CancelReplay
	Ereignisse: CreateApi Reiseziel
	Ereignisse: CreateArchive
	Ereignisse: CreateConnection
	Ereignisse: CreateEndpoint
	Veranstaltungen: CreateEvent Bus
	Ereignisse: CreatePartner EventSource
	Ereignisse: DeactivateEvent Quelle
	Ereignisse: DeauthorizeConnection
	Ereignisse: DeleteApi Reiseziel
	Ereignisse: DeleteArchive
	Ereignisse: DeleteConnection
	Ereignisse: DeleteEndpoint
	Veranstaltungen: DeleteEvent Bus
	Ereignisse: DeletePartner EventSource
	Ereignisse: DeleteRule
	Ereignisse: DescribeApi Reiseziel
	Ereignisse: DescribeArchive
	Ereignisse: DescribeConnection
	Ereignisse: DescribeEndpoint

Servicepräfix	Aktionen
	Veranstaltungen: DescribeEvent Bus
	Ereignisse: DescribeEvent Quelle
	Ereignisse: DescribePartner EventSource
	Ereignisse: DescribeReplay
	Ereignisse: DescribeRule
	Ereignisse: DisableRule
	Ereignisse: EnableRule
	Ereignisse: ListApi Destinationen
	Ereignisse: ListArchives
	Ereignisse: ListConnections
	Ereignisse: ListEndpoints
	Veranstaltungen: ListEvent Busse
	Ereignisse: ListEvent Quellen
	Ereignisse: ListPartner EventSource Konten
	Ereignisse: ListPartner EventSources
	Ereignisse: ListReplays
	Ereignisse: ListRule NamesBy Ziel
	Ereignisse: ListRules
	Ereignisse: ListTargets ByRule
	Ereignisse: PutPermission
	Ereignisse: PutRule

Servicepräfix	Aktionen
	Ereignisse: PutTargets
	Ereignisse: RemovePermission
	Ereignisse: RemoveTargets
	Ereignisse: StartReplay
	Ereignisse: TestEvent Muster
	Ereignisse: UpdateApi Ziel
	Ereignisse: UpdateArchive
	Ereignisse: UpdateConnection
	Ereignisse: UpdateEndpoint

Servicepräfix	Aktionen
evidently	offensichtlich: CreateExperiment offensichtlich: CreateFeature offensichtlich: CreateLaunch offensichtlich: CreateProject offensichtlich: CreateSegment offensichtlich: DeleteExperiment offensichtlich: DeleteFeature offensichtlich: DeleteLaunch offensichtlich: DeleteProject offensichtlich: DeleteSegment offensichtlich: GetExperiment offensichtlich: Ergebnisse GetExperiment offensichtlich: GetFeature offensichtlich: GetLaunch offensichtlich: GetProject offensichtlich: GetSegment offensichtlich: ListExperiments offensichtlich: ListFeatures offensichtlich: ListLaunches offensichtlich: ListProjects offenbar: Referenzen ListSegment

Servicepräfix	Aktionen
	offensichtlich: ListSegments offensichtlich: StartExperiment offensichtlich: StartLaunch offensichtlich: StopExperiment offensichtlich: StopLaunch offenbar: Muster TestSegment offensichtlich: UpdateExperiment offensichtlich: UpdateFeature offensichtlich: UpdateLaunch offensichtlich: UpdateProject offensichtlich: UpdateProject DataDelivery

Servicepräfix	Aktionen
finspace	Finspace: CreateEnvironment finspace: Änderungssatz CreateKx finspace: Cluster CreateKx finspace: Datenbank CreateKx finspace: Datenansicht CreateKx finspace: Umgebung CreateKx finspace: CreateKx ScalingGroup finspace: Benutzer CreateKx finspace: Volumen CreateKx finspace: CreateUser Finspace: DeleteEnvironment finspace: Cluster DeleteKx finspace: DeleteKx ClusterNode finspace: Datenbank DeleteKx finspace: Datenansicht DeleteKx finspace: Umgebung DeleteKx finspace: DeleteKx ScalingGroup finspace: Benutzer DeleteKx finspace: Volumen DeleteKx finspace: GetEnvironment finspace: Änderungssatz GetKx

Servicepräfix	Aktionen
	finspace: Cluster GetKx
	finspace: GetKx ConnectionString
	finspace: Datenbank GetKx
	finspace: Datenansicht GetKx
	finspace: Umgebung GetKx
	finspace: GetKx ScalingGroup
	finspace: Benutzer GetKx
	finspace: Volumen GetKx
	finspace: Status GetLoad SampleData SetGroup IntoEnvironment
	finspace: GetUser
	Finspace: ListEnvironments
	finspace: Änderungssätze ListKx
	finspace: ListKx ClusterNodes
	finspace: Cluster ListKx
	finspace: Datenbanken ListKx
	finspace: Datenansichten ListKx
	ListKxfinspace: Umgebungen
	finspace: ListKx ScalingGroups
	finspace: Benutzer ListKx
	finspace: Volumen ListKx
	finspace: ListUsers

Servicepräfix	Aktionen
	finspace: Umwelt LoadSample DataSet GroupInto finspace: Passwort ResetUser finspace: UpdateEnvironment finspace: Konfiguration UpdateKx ClusterCode finspace: UpdateKx ClusterDatabases finspace: Datenbank UpdateKx finspace: Datenansicht UpdateKx finspace: Umgebung UpdateKx finspace: UpdateKx EnvironmentNetwork finspace: Benutzer UpdateKx finspace: Volumen UpdateKx finspace: UpdateUser
firehose	Firehose: Stream CreateDelivery Firehose: Bach DeleteDelivery Firehose: Bach DescribeDelivery Firehose: Streams ListDelivery Feuerwehrschlauch: StartDelivery StreamEncryption Feuerwehrschlauch: StopDelivery StreamEncryption Feuerwehrschlauch: UpdateDestination

Servicepräfix	Aktionen
fis	fis: Vorlage CreateExperiment fis: CreateTarget AccountConfiguration fis: Vorlage DeleteExperiment fis: DeleteTarget AccountConfiguration passt: GetAction passt: GetExperiment fis: Konfiguration GetExperiment TargetAccount fis: Vorlage GetExperiment fis: GetTarget AccountConfiguration passt: GetTarget ResourceType passt: ListActions passt: ListExperiment ResolvedTargets passt: ListExperiments fis: Konfigurationen ListExperiment TargetAccount fis: Vorlagen ListExperiment fis: ListTarget AccountConfigurations passt: ListTarget ResourceTypes passt: StartExperiment passt: StopExperiment fis: Vorlage UpdateExperiment fis: UpdateTarget AccountConfiguration

Servicepräfix	Aktionen
fms	fms: Konto AssociateAdmin fms: AssociateThird PartyFirewall fms: Ressource BatchAssociate fms: Ressource BatchDisassociate fms: Liste DeleteApps fms: Kanal DeleteNotification fms: DeletePolicy fms: Liste DeleteProtocols fms: Satz DeleteResource fms: Konto DisassociateAdmin fms: DisassociateThird PartyFirewall fms: Konto GetAdmin fms: Geltungsbereich GetAdmin fms: Liste GetApps fms: Detail GetCompliance fms: Kanal GetNotification fms: GetPolicy fms: Status GetProtection fms: Liste GetProtocols fms: Satz GetResource fms: GetThird PartyFirewall AssociationStatus

Servicepräfix	Aktionen
	fms: Einzelheiten GetViolation fms: Organisation ListAdmin AccountsFor fms: ListAdmins ManagingAccount fms: Listen ListApps fms: Status ListCompliance fms: Ressourcen ListDiscovered fms: Konten ListMember fms: ListPolicies fms: Listen ListProtocols fms: ListResource SetResources fms: Sätze ListResource fms: ListThird PartyFirewall FirewallPolicies fms: Konto PutAdmin fms: Liste PutApps fms: Kanal PutNotification fms: PutPolicy fms: Liste PutProtocols fms: Satz PutResource

Servicepräfix	Aktionen
frauddetector	Betrugsdetektor: Variabel BatchCreate
	Betrugsdetektor: Variabel BatchGet
	Betrugsdetektor: CancelBatch ImportJob
	Betrugsdetektor: CancelBatch PredictionJob
	Betrugsdetektor: CreateBatch ImportJob
	Betrugsdetektor: CreateBatch PredictionJob
	Betrugsdetektor: Version CreateDetector
	Betrugsdetektor: CreateList
	Betrugsdetektor: CreateModel
	Betrugsdetektor: Version CreateModel
	Betrugsdetektor: CreateRule
	Betrugsdetektor: CreateVariable
	Betrugsdetektor: DeleteBatch ImportJob
	Betrugsdetektor: DeleteBatch PredictionJob
	Betrugsdetektor: DeleteDetector
	Betrugsdetektor: Version DeleteDetector
	Frauddetector: Typ DeleteEntity
	Betrugsdetektor: DeleteEvent
	Betrugsdetektor: Typ DeleteEvents ByEvent
	Frauddetector: Typ DeleteEvent
	Frauddetector: Modell DeleteExternal

Servicepräfix	Aktionen
	Betrugsdetektor: DeleteLabel
	Betrugsdetektor: DeleteList
	Betrugsdetektor: DeleteModel
	Betrugsdetektor: Version DeleteModel
	Betrugsdetektor: DeleteOutcome
	Betrugsdetektor: DeleteRule
	Betrugsdetektor: DeleteVariable
	Betrugsdetektor: DescribeDetector
	frauddetector: Versionen DescribeModel
	Betrugsdetektor: GetBatch ImportJobs
	Betrugsdetektor: GetBatch PredictionJobs
	Betrugsdetektor: Status GetDelete EventsBy EventType
	Betrugsdetektor: GetDetectors
	Betrugsdetektor: Version GetDetector
	Frauddetector: Typen GetEntity
	Betrugsdetektor: GetEvent
	Frauddetector: Vorhersage GetEvent
	Betrugsdetektor: GetEvent PredictionMetadata
	Betrugsdetektor: Typen GetEvent
	Frauddetector: Modelle GetExternal
	Betrugsdetektor: GetKMS EncryptionKey

Servicepräfix	Aktionen
	Betrugsdetektor: GetLabels
	Betrugsdetektor: Elemente GetList
	Frauddetector: Metadaten GetLists
	Betrugsdetektor: GetModels
	Betrugsdetektor: Version GetModel
	Betrugsdetektor: GetOutcomes
	Betrugsdetektor: GetRules
	Betrugsdetektor: GetVariables
	Frauddetector: Prognosen ListEvent
	Betrugsdetektor: PutDetector
	Betrugsdetektor: Typ PutEntity
	Frauddetector: Typ PutEvent
	Frauddetector: Modell PutExternal
	Betrugsdetektor: PutKMS EncryptionKey
	Betrugsdetektor: PutLabel
	Betrugsdetektor: PutOutcome
	Betrugsdetektor: SendEvent
	Betrugsdetektor: Version UpdateDetector
	Betrugsdetektor: UpdateDetector VersionMetadata
	Betrugsdetektor: UpdateDetector VersionStatus
	Frauddetector: Etikett UpdateEvent

Servicepräfix	Aktionen
	Betrugsdetektor: UpdateList Betrugsdetektor: UpdateModel Betrugsdetektor: Version UpdateModel Betrugsdetektor: UpdateModel VersionStatus frauddetector: Metadaten UpdateRule Frauddetector: Version UpdateRule Betrugsdetektor: UpdateVariable

Servicepräfix	Aktionen
fsx	fax: AssociateFile SystemAliases fsx: CancelData RepositoryTask fsx: CopyBackup fsx: CreateData RepositoryTask fsx: Zwischenspeicher CreateFile fsx: System CreateFile fsx: Backup CreateFile SystemFrom fsx: CreateSnapshot fsx: CreateStorage VirtualMachine fsx: CreateVolume fsx: CreateVolume FromBackup fsx: DeleteBackup fsx: Zwischenspeicher DeleteFile fsx: System DeleteFile fsx: DeleteSnapshot fsx: DeleteStorage VirtualMachine fsx: DeleteVolume fsx: DescribeBackups fsx: DescribeData RepositoryAssociations fsx: DescribeData RepositoryTasks fsx: Zwischenspeicher DescribeFile

Servicepräfix	Aktionen
	fsx: DescribeFile SystemAliases fsx: Systeme DescribeFile fsx: DescribeShared VpcConfiguration fsx: DescribeSnapshots fsx: DescribeStorage VirtualMachines fsx: DescribeVolumes fsx: DisassociateFile SystemAliases fsx: V3Locks ReleaseFile SystemNfs fsx: RestoreVolume FromSnapshot fsx: StartMisconfigured StateRecovery fsx: UpdateData RepositoryAssociation fsx: Zwischenspeicher UpdateFile fsx: System UpdateFile fsx: UpdateShared VpcConfiguration fsx: UpdateSnapshot fsx: UpdateStorage VirtualMachine fsx: UpdateVolume

Servicepräfix	Aktionen
gamelift	Gamelift: AcceptMatch
	Gamelift: Server ClaimGame
	Gamelift: CreateAlias
	Gamelift: CreateBuild
	Gamelift: CreateContainer GroupDefinition
	Gamelift: CreateFleet
	Gamelift: Standorte CreateFleet
	Gamelift: CreateGame ServerGroup
	Gamelift: Sitzung CreateGame
	Gamelift: CreateGame SessionQueue
	Gamelift: CreateLocation
	Gamelift: Konfiguration CreateMatchmaking
	Gamelift: CreateMatchmaking RuleSet
	Gamelift: Sitzung CreatePlayer
	Gamelift: Sitzungen CreatePlayer
	Gamelift: CreateScript
	Gamelift: CreateVpc PeeringAuthorization
	Gamelift: CreateVpc PeeringConnection
	Gamelift: DeleteAlias
	Gamelift: DeleteBuild
	Gamelift: DeleteContainer GroupDefinition

Servicepräfix	Aktionen
	Gamelift: DeleteFleet
	Gamelift: Standorte DeleteFleet
	Gamelift: DeleteGame ServerGroup
	Gamelift: DeleteGame SessionQueue
	Gamelift: DeleteLocation
	Gamelift: Konfiguration DeleteMatchmaking
	Gamelift: DeleteMatchmaking RuleSet
	Gamelift: Richtlinie DeleteScaling
	Gamelift: DeleteScript
	Gamelift: DeleteVpc PeeringAuthorization
	Gamelift: DeleteVpc PeeringConnection
	Gamelift: DeregisterCompute
	Gamelift: Server DeregisterGame
	Gamelift: DescribeAlias
	Gamelift: DescribeBuild
	Gamelift: DescribeCompute
	Gamelift: DescribeContainer GroupDefinition
	Gamelift: Beschreibe EC2 InstanceLimits
	gamelift: Attribute DescribeFleet
	Gamelift: Kapazität DescribeFleet
	Gamelift: Veranstaltungen DescribeFleet

Servicepräfix	Aktionen
	Gamelift: DescribeFleet LocationAttributes
	Gamelift: DescribeFleet LocationCapacity
	Gamelift: DescribeFleet LocationUtilization
	Gamelift: DescribeFleet PortSettings
	Gamelift: Nutzung DescribeFleet
	Gamelift: Server DescribeGame
	Gamelift: DescribeGame ServerGroup
	Gamelift: DescribeGame ServerInstances
	Gamelift: DescribeGame SessionDetails
	Gamelift: DescribeGame SessionPlacement
	Gamelift: DescribeGame SessionQueues
	Gamelift: Sitzungen DescribeGame
	Gamelift: DescribeInstances
	Gamelift: DescribeMatchmaking
	Gamelift: Konfigurationen DescribeMatchmaking
	Gamelift: DescribeMatchmaking RuleSets
	Gamelift: Sitzungen DescribePlayer
	Gamelift: Konfiguration DescribeRuntime
	Gamelift: Richtlinien DescribeScaling
	Gamelift: DescribeScript
	Gamelift: DescribeVpc PeeringAuthorizations

Servicepräfix	Aktionen
	Gamelift: DescribeVpc PeeringConnections
	Gamelift: Zugriff GetCompute
	Gamelift: GetCompute AuthToken
	Gamelift: URL GetGame SessionLog
	Gamelift: Zugriff GetInstance
	Gamelift: ListAliases
	Gamelift: ListBuilds
	Gamelift: ListCompute
	Gamelift: ListContainer GroupDefinitions
	Gamelift: ListFleets
	Gamelift: ListGame ServerGroups
	Gamelift: Server ListGame
	Gamelift: ListLocations
	Gamelift: ListScripts
	Gamelift: Richtlinie PutScaling
	Gamelift: RegisterCompute
	Gamelift: Server RegisterGame
	gamelift: Zugangsdaten RequestUpload
	Gamelift: ResolveAlias
	Gamelift: ResumeGame ServerGroup
	Gamelift: Sitzungen SearchGame

Servicepräfix	Aktionen
	Gamelift: Aktionen StartFleet
	Gamelift: StartGame SessionPlacement
	Gamelift: Auffüllen StartMatch
	Gamelift: StartMatchmaking
	Gamelift: Aktionen StopFleet
	Gamelift: StopGame SessionPlacement
	Gamelift: StopMatchmaking
	Gamelift: SuspendGame ServerGroup
	Gamelift: UpdateAlias
	Gamelift: UpdateBuild
	Gamelift: Attribute UpdateFleet
	Gamelift: Kapazität UpdateFleet
	Gamelift: UpdateFleet PortSettings
	Gamelift: Server UpdateGame
	Gamelift: UpdateGame ServerGroup
	Gamelift: Sitzung UpdateGame
	Gamelift: UpdateGame SessionQueue
	Gamelift: Konfiguration UpdateMatchmaking
	Gamelift: Konfiguration UpdateRuntime
	Gamelift: UpdateScript
	Gamelift: ValidateMatchmaking RuleSet

Servicepräfix	Aktionen
geo	geo: Verbraucher AssociateTracker geo: BatchDelete DevicePosition Geschichte geo: BatchDelete Geofence geo: Geofences BatchEvaluate Geo: BatchGet DevicePosition geo: BatchPut Geofence Geo: BatchUpdate DevicePosition Geo: CalculateRoute geo: CalculateRoute Matrix geo: CreateGeofence Sammlung geo: CreateMap geo: CreatePlace Index geo: CreateRoute Rechner geo: CreateTracker geo: DeleteGeofence Sammlung geo: DeleteKey Geo: DeleteMap geo: DeletePlace Index geo: DeleteRoute Rechner geo: DeleteTracker geo: DescribeGeofence Sammlung

Servicepräfix	Aktionen
	geo: DescribeKey Geo: DescribeMap geo: DescribePlace Index geo: DescribeRoute Rechner geo: DescribeTracker geo: DisassociateTracker Verbraucher geo: GetDevice Position Geo: GetDevice PositionHistory Geo: GetGeofence geo: GetMap Glyphen geo: Sprites GetMap Geo: GetMap StyleDescriptor geo: GetMap Titel geo: GetPlace geo: ListDevice Positionen geo: ListGeofence Sammlungen geo: ListGeofences Geo: ListKeys Geo: ListMaps geo: ListPlace Indizes geo: Taschenrechner ListRoute

Servicepräfix	Aktionen
	geo: Verbraucher ListTracker geo: ListTrackers Geo: PutGeofence geo: SearchPlace IndexFor Position geo: SearchPlace IndexFor Vorschläge geo: SearchPlace IndexFor Text geo: UpdateGeofence Sammlung geo: UpdateKey Geo: UpdateMap geo: UpdatePlace Index geo: UpdateRoute Rechner geo: UpdateTracker

Servicepräfix	Aktionen
glacier	Gletscher: AbortMultipart Hochladen
	Gletscher: AbortVault Sperren
	Gletscher: CompleteMultipart Hochladen
	Gletscher: CompleteVault Sperren
	Gletscher: CreateVault
	Gletscher: DeleteArchive
	Gletscher: DeleteVault
	Gletscher: DeleteVault AccessPolicy
	Gletscher: DeleteVault Benachrichtigungen
	Gletscher: DescribeJob
	Gletscher: DescribeVault
	Gletscher: GetData RetrievalPolicy
	Gletscher: GetJob Ausgabe
	Gletscher: GetVault AccessPolicy
	Gletscher: GetVault Schleuse
	Glacier: GetVault Benachrichtigungen
	Gletscher: InitiateJob
	Gletscher: InitiateMultipart Hochladen
	Gletscher: InitiateVault Sperren
	Gletscher: ListJobs
	Gletscher: ListMultipart Hochgeladene Dateien

Servicepräfix	Aktionen
	Gletscher: ListParts
	Gletscher: ListProvisioned Kapazität
	Gletscher: ListVaults
	Gletscher: PurchaseProvisioned Kapazität
	Gletscher: SetData RetrievalPolicy
	Gletscher: SetVault AccessPolicy
	Gletscher: SetVault Benachrichtigungen
	Gletscher: UploadArchive
	Gletscher: UploadMultipart Teil

Servicepräfix	Aktionen
grafana	Grafana: AssociateLicense Grafana: CreateWorkspace Grafana: CreateWorkspace ApiKey Grafana: DeleteWorkspace Grafana: DeleteWorkspace ApiKey Grafana: DescribeWorkspace grafana: Authentifizierung DescribeWorkspace grafana: Konfiguration DescribeWorkspace grafana: DisassociateLicense Grafana: ListPermissions Grafana: ListVersions Grafana: ListWorkspaces Grafana: UpdatePermissions Grafana: UpdateWorkspace grafana: Authentifizierung UpdateWorkspace grafana: Konfiguration UpdateWorkspace

Servicepräfix	Aktionen
greengrass	grünes Gras: AssociateRole ToGroup
	greengrass: Konto AssociateService RoleTo
	greengrass: Gerät BatchAssociate ClientDevice WithCore
	greengrass: Gerät BatchDisassociate ClientDevice FromCore
	greengrass: CancelDeployment
	Greengrass: Ausführung CreateComponent
	Grüngras: Definition CreateConnector
	Grüngras: CreateConnector DefinitionVersion
	Grüngras: Definition CreateCore
	Grüngras: CreateCore DefinitionVersion
	grünes Gras: CreateDeployment
	Grüngras: Definition CreateDevice
	Grüngras: CreateDevice DefinitionVersion
	Grüngras: Definition CreateFunction
	Grüngras: CreateFunction DefinitionVersion
	grünes Gras: CreateGroup
	grünes Gras: CreateGroup CertificateAuthority
	Greengrass: Ausführung CreateGroup
	Grüngras: Definition CreateLogger
	Grüngras: CreateLogger DefinitionVersion
	Grüngras: Definition CreateResource

Servicepräfix	Aktionen
	Grüngras: CreateResource DefinitionVersion
	grünes Gras: CreateSoftware UpdateJob
	Grüngras: Definition CreateSubscription
	Grüngras: CreateSubscription DefinitionVersion
	grünes Gras: DeleteComponent
	Grüngras: Definition DeleteConnector
	Grüngras: Definition DeleteCore
	greengrass: Gerät DeleteCore
	greengrass: DeleteDeployment
	Grüngras: Definition DeleteDevice
	Grüngras: Definition DeleteFunction
	Grüngras: DeleteGroup
	Grüngras: Definition DeleteLogger
	Grüngras: Definition DeleteResource
	Grüngras: Definition DeleteSubscription
	Grüngras: DescribeComponent
	grünes Gras: DisassociateRole FromGroup
	greengrass: Konto DisassociateService RoleFrom
	greengrass: Rolle GetAssociated
	Grüngras: GetBulk DeploymentStatus
	grünes Gras: GetComponent

Servicepräfix	Aktionen
	grünes Gras: GetComponent VersionArtifact
	Grüngras: Informationen GetConnectivity
	Grüngras: Definition GetConnector
	Grüngras: GetConnector DefinitionVersion
	Grüngras: Definition GetCore
	Grüngras: GetCore DefinitionVersion
	greengrass: Gerät GetCore
	greengrass: GetDeployment
	Grüngras: Status GetDeployment
	Grüngras: Definition GetDevice
	Grüngras: GetDevice DefinitionVersion
	Grüngras: Definition GetFunction
	Grüngras: GetFunction DefinitionVersion
	grünes Gras: GetGroup
	grünes Gras: GetGroup CertificateAuthority
	grünes Gras: GetGroup CertificateConfiguration
	Greengrass: Ausführung GetGroup
	Grüngras: Definition GetLogger
	Grüngras: GetLogger DefinitionVersion
	Grüngras: Definition GetResource
	Grüngras: GetResource DefinitionVersion

Servicepräfix	Aktionen
	greengrass: Konto GetService RoleFor
	greengrass: Definition GetSubscription
	Grüngras: GetSubscription DefinitionVersion
	grünes Gras: GetThing RuntimeConfiguration
	greengrass: Berichte ListBulk DeploymentDetailed
	greengrass: Bereitstellungen ListBulk
	greengrass: Gerät ListClient DevicesAssociated WithCore
	greengrass: ListComponents
	greengrass: Versionen ListComponent
	greengrass: Definitionen ListConnector
	Grüngras: ListConnector DefinitionVersions
	Greengrass: Definitionen ListCore
	Grüngras: ListCore DefinitionVersions
	greengrass: Geräte ListCore
	greengrass: ListDeployments
	Greengrass: Definitionen ListDevice
	Grüngras: ListDevice DefinitionVersions
	greengrass: Bereitstellungen ListEffective
	greengrass: Definitionen ListFunction
	Grüngras: ListFunction DefinitionVersions
	grünes Gras: ListGroup CertificateAuthorities

Servicepräfix	Aktionen
	grünes Gras: ListGroups
	greengrass: Versionen ListGroup
	greengrass: Komponenten ListInstalled
	Greengrass: Definitionen ListLogger
	Grüngras: ListLogger DefinitionVersions
	Greengrass: Definitionen ListResource
	Grüngras: ListResource DefinitionVersions
	Greengrass: Definitionen ListSubscription
	Grüngras: ListSubscription DefinitionVersions
	grünes Gras: ResetDeployments
	greengrass: Einsatz StartBulk
	greengrass: Einsatz StopBulk
	greengrass: Informationen UpdateConnectivity
	Grüngras: Definition UpdateConnector
	Grüngras: Definition UpdateCore
	Grüngras: Definition UpdateDevice
	Grüngras: Definition UpdateFunction
	Grüngras: UpdateGroup
	grünes Gras: UpdateGroup CertificateConfiguration
	Grüngras: Definition UpdateLogger
	Grüngras: Definition UpdateResource

Servicepräfix	Aktionen
	Grüngras: Definition UpdateSubscription Grüngras: UpdateThing RuntimeConfiguration

Servicepräfix	Aktionen
groundstation	Bodenstation: CancelContact Bodenstation: CreateConfig Bodenstation: CreateDataflow EndpointGroup Bodenstation: CreateEphemeris Bodenstation: Profil CreateMission Bodenstation: DeleteConfig Bodenstation: DeleteDataflow EndpointGroup Bodenstation: DeleteEphemeris Bodenstation: Profil DeleteMission Bodenstation: DescribeContact Bodenstation: DescribeEphemeris Bodenstation: GetConfig Bodenstation: GetDataflow EndpointGroup Bodenstation: Nutzung GetMinute Bodenstation: Profil GetMission Bodenstation: GetSatellite Bodenstation: ListConfigs Bodenstation: ListContacts Bodenstation: ListDataflow EndpointGroups Bodenstation: ListEphemerides Bodenstation: Stationen ListGround

Servicepräfix	Aktionen
	Bodenstation: Profile ListMission Bodenstation: ListSatellites Bodenstation: RegisterAgent Bodenstation: ReserveContact Bodenstation: Status UpdateAgent Bodenstation: UpdateConfig Bodenstation: UpdateEphemeris Bodenstation: Profil UpdateMission

Servicepräfix	Aktionen
guardduty	guardduty: Einladung AcceptAdministrator
	Guardduty: AcceptInvitation
	Wachdienst: ArchiveFindings
	Wachdienst: CreateDetector
	Wachdienst: CreateFilter
	guardduty:CreateIPSet
	Wachdienst: CreateMembers
	guardduty: Ziel CreatePublishing
	guardduty: Ergebnisse CreateSample
	Guardduty: CreateThreat IntelSet
	Wachdienst: DeclineInvitations
	Wachdienst: DeleteDetector
	Wachdienst: DeleteFilter
	Wachdienst: DeleteInvitations
	guardduty:DeleteIPSet
	Wachdienst: DeleteMembers
	guardduty: Ziel DeletePublishing
	Guardduty: DeleteThreat IntelSet
	guardduty: Scannt DescribeMalware
	guardduty: Konfiguration DescribeOrganization
	guardduty: Ziel DescribePublishing

Servicepräfix	Aktionen
	Guardduty: DisableOrganization AdminAccount
	Wachdienst: DisassociateFrom AdministratorAccount
	Wachdienst: DisassociateFrom MasterAccount
	Wachdienst: DisassociateMembers
	Wachdienst: EnableOrganization AdminAccount
	guardduty: Konto GetAdministrator
	guardduty: Statistiken GetCoverage
	Guardduty: GetDetector
	Wachdienst: GetFilter
	Wachdienst: GetFindings
	guardduty: Statistiken GetFindings
	guardduty: Zähle GetInvitations
	guardduty:GetIPSet
	Guardduty: GetMalware ScanSettings
	guardduty: Konto GetMaster
	guardduty: Detektoren GetMember
	Wachdienst: GetMembers
	guardduty: Statistiken GetOrganization
	Guardduty: Tage GetRemaining FreeTrial
	Guardduty: GetThreat IntelSet
	guardduty: Statistiken GetUsage

Servicepräfix	Aktionen
	Guardduty: InviteMembers
	Wachdienst: ListCoverage
	Wachdienst: ListDetectors
	Wachdienst: ListFilters
	Wachdienst: ListFindings
	Wachdienst: ListInvitations
	guardduty:ListIPSets
	Wachdienst: ListMembers
	Wachdienst: ListOrganization AdminAccounts
	guardduty: Ziele ListPublishing
	Guardduty: ListThreat IntelSets
	guardduty: Telemetrie SendSecurity
	guardduty: Scannen StartMalware
	guardduty: Mitglieder StartMonitoring
	guardduty: Mitglieder StopMonitoring
	guardduty: UnarchiveFindings
	Wachdienst: UpdateDetector
	Wachdienst: UpdateFilter
	guardduty: Rückmeldung UpdateFindings
	guardduty:UpdateIPSet
	Guardduty: UpdateMalware ScanSettings

Servicepräfix	Aktionen
	guardduty: Detektoren UpdateMember guardduty: Konfiguration UpdateOrganization guardduty: Ziel UpdatePublishing Guardduty: UpdateThreat IntelSet
healthlake	healthlake:CreateFHIRDatastore Gesundheit, See: CreateResource healthlake:DeleteFHIRDatastore Healthlake: DeleteResource healthlake:DescribeFHIRDatastore Healthlake: Beschreiben Sie FHIR ExportJob Health Lake: Beschreibe Fhir ImportJob Healthlake: GetCapabilities healthlake:ListFHIRDatastores Healthlake: Liste FHIR ExportJobs Gesundheitssee: Listfhir ImportJobs Gesundheitssee: ReadResource Healthlake: Holen SearchWith healthlake: Beitrag SearchWith Healthlake: FHIR starten ExportJob Gesundheitssee: Startfhir ImportJob Gesundheitssee: UpdateResource

Servicepräfix	Aktionen
honeycode	Honeycode: BatchCreate TableRows Honigcode: BatchDelete TableRows Honigcode: BatchUpdate TableRows Honigcode: BatchUpsert TableRows honeycode: Job DescribeTable DataImport honeycode: Daten GetScreen honeycode: Automatisierung InvokeScreen honeycode: Spalten ListTable honeycode: Zeilen ListTable Honeycode: ListTables honeycode: Zeilen QueryTable honeycode: Job StartTable DataImport

Servicepräfix	Aktionen
iam	iam: ID ID AddClient ToOpen ConnectProvider iam: Profil AddRole ToInstance iam: AddUser ToGroup iam: Richtlinie AttachGroup iam: Richtlinie AttachRole iam: Richtlinie AttachUser ich bin: ChangePassword iam: Schlüssel CreateAccess iam: Alias CreateAccount ich bin: CreateGroup iam: Profil CreateInstance iam: Profil CreateLogin iam: ID CreateOpen ConnectProvider ich bin: CreatePolicy iam: Version CreatePolicy ich bin: CreateRole iam:CreateSAMLProvider ich bin: CreateService LinkedRole ich bin: CreateService SpecificCredential ich bin: CreateUser Ich bin: MFADevice CreateVirtual

Servicepräfix	Aktionen
	iam:DeactivateMFADevice iam: Schlüssel DeleteAccess iam: Alias DeleteAccount ich bin: DeleteAccount PasswordPolicy iam: Schlüssel DeleteCloud FrontPublic ich bin: DeleteGroup iam: Richtlinie DeleteGroup iam: Profil DeleteInstance iam: Profil DeleteLogin iam: ID DeleteOpen ConnectProvider ich bin: DeletePolicy iam: Version DeletePolicy ich bin: DeleteRole ich bin: DeleteRole PermissionsBoundary iam: Richtlinie DeleteRole iam:DeleteSAMLProvider iam: Zertifikat DeleteServer iam: DeleteService LinkedRole ich bin: DeleteService SpecificCredential iam: Zertifikat DeleteSigning iam: Löscht SSH PublicKey

Servicepräfix	Aktionen
	ich bin: DeleteUser ich bin: DeleteUser PermissionsBoundary iam: Richtlinie DeleteUser Ich bin: MFA-Gerät DeleteVirtual iam: Richtlinie DetachGroup iam: Richtlinie DetachRole iam: Richtlinie DetachUser iam:EnableMFADevice iam: Bericht GenerateCredential ich bin: GenerateOrganizations AccessReport iam: Einzelheiten GenerateService LastAccessed iam: Benutzt GetAccess KeyLast ich bin: GetAccount AuthorizationDetails ich bin: GetAccount EmailAddress iam: Name GetAccount ich bin: GetAccount PasswordPolicy iam: Zusammenfassung GetAccount iam: Schlüssel GetCloud FrontPublic ich bin: GetContext KeysFor CustomPolicy ich bin: GetContext KeysFor PrincipalPolicy iam: Bericht GetCredential

Servicepräfix	Aktionen
	ich bin: GetGroup iam: Richtlinie GetGroup iam: Profil GetInstance iam: Profil GetLogin iam:GetMFADevice iam: ID GetOpen ConnectProvider ich bin: GetOrganizations AccessReport ich bin: GetPolicy iam: Version GetPolicy ich bin: GetRole iam: Richtlinie GetRole iam:GetSAMLProvider iam: Zertifikat GetServer iam: Einzelheiten GetService LastAccessed iam: Entitäten GetService LastAccessed DetailsWith iam: GetService LinkedRole DeletionStatus Ich bin: bekommt SSH PublicKey ich bin: GetUser iam: Richtlinie GetUser iam: Schlüssel ListAccess iam: Aliase ListAccount

Servicepräfix	Aktionen
	ich bin: ListAttached GroupPolicies ich bin: ListAttached RolePolicies ich bin: ListAttached UserPolicies iam: Schlüssel ListCloud FrontPublic ich bin: ListEntities ForPolicy iam: Richtlinien ListGroup iam: ListGroups ich bin: ListGroups ForUser iam: Profile ListInstance iam: Rolle ListInstance ProfilesFor iam:ListMFADevices iam: ID ListOpen ConnectProviders ich bin: ListPolicies iam: Zugriff ListPolicies GrantingService iam: Versionen ListPolicy iam: Richtlinien ListRole iam: ListRoles iam:ListSAMLProviders iam: Zertifikate ListServer iam: ListService SpecificCredentials iam: Zertifikate ListSigning

Servicepräfix	Aktionen
	iam: ListSSH PublicKeys Ich bin: ListSTS-Status RegionalEndpoints iam: Richtlinien ListUser iam: ListUsers Ich bin: MFA-Geräte ListVirtual iam: Richtlinie PutGroup ich bin: PutRole PermissionsBoundary iam: Richtlinie PutRole ich bin: PutUser PermissionsBoundary iam: Richtlinie PutUser iam: ID ID RemoveClient FromOpen ConnectProvider iam: Profil RemoveRole FromInstance iam: RemoveUser FromGroup ich bin: ResetService SpecificCredential iam:ResyncMFADevice ich bin: SetDefault PolicyVersion iam: Einstellungen SetSecurity TokenService RegionalEndpointiam: Legt den STS-Status fest iam: Richtlinie SimulateCustom iam: Richtlinie SimulatePrincipal iam: Schlüssel UpdateAccess

Servicepräfix	Aktionen
	ich bin: UpdateAccount EmailAddress iam: Name UpdateAccount ich bin: UpdateAccount PasswordPolicy ich bin: UpdateAssume RolePolicy iam: Schlüssel UpdateCloud FrontPublic ich bin: UpdateGroup iam: Profil UpdateLogin iam: UpdateOpen ID Thumbprint ConnectProvider ich bin: UpdateRole iam: Beschreibung UpdateRole iam:UpdateSAMLProvider iam: Zertifikat UpdateServer iam: UpdateService SpecificCredential iam: Zertifikat UpdateSigning iam: Aktualisiert SSH PublicKey ich bin: UpdateUser iam: Schlüssel UploadCloud FrontPublic iam: Zertifikat UploadServer iam: Zertifikat UploadSigning iam: lädt SSH hoch PublicKey

Servicepräfix	Aktionen
identitystore	Identitätsspeicher: CreateGroup identitystore: Mitgliedschaft CreateGroup Identitystore: CreateUser Identitätsspeicher: DeleteGroup identitystore: Mitgliedschaft DeleteGroup Identitystore: DeleteUser Identitätsspeicher: DescribeGroup identitystore: Mitgliedschaft DescribeGroup Identitystore: DescribeUser Identitystore: ID GetGroup Identitätsspeicher: GetGroup MembershipId Identitystore: ID GetUser Identitätsspeicher: IsMember InGroups identitystore: Mitgliedschaften ListGroup identitystore: ListGroup MembershipsFor Mitglied identitystore: ListGroups Identitätsspeicher: ListUsers Identitätsspeicher: UpdateGroup Identitätsspeicher: UpdateUser

Servicepräfix	Aktionen
imagebuilder	imagebuilder: Schöpfung CancelImage imagebuilder: Ausführung CancelLifecycle Imagebuilder: CreateComponent imagebuilder: Rezept CreateContainer imagebuilder: Konfiguration CreateDistribution imagebuilder: CreateImage imagebuilder: Pipeline CreateImage imagebuilder: Rezept CreateImage imagebuilder: Konfiguration CreateInfrastructure imagebuilder: Richtlinie CreateLifecycle imagebuilder: CreateWorkflow Image-Builder: DeleteComponent imagebuilder: Rezept DeleteContainer imagebuilder: Konfiguration DeleteDistribution imagebuilder: DeleteImage imagebuilder: Pipeline DeleteImage imagebuilder: Rezept DeleteImage imagebuilder: Konfiguration DeleteInfrastructure imagebuilder: Richtlinie DeleteLifecycle imagebuilder: DeleteWorkflow imagebuilder: Richtlinie GetComponent

Servicepräfix	Aktionen
	imagebuilder: GetContainer RecipePolicy imagebuilder: Richtlinie GetImage imagebuilder: GetImage RecipePolicy imagebuilder: Ausführung GetLifecycle imagebuilder: Richtlinie GetLifecycle imagebuilder: Ausführung GetWorkflow Imagebuilder: GetWorkflow StepExecution Bildgenerator: ImportComponent imagebuilder: Bild ImportVm Imagebuilder: ListComponent BuildVersions Bildgenerator: ListComponents imagebuilder: Rezepte ListContainer imagebuilder: Konfigurationen ListDistribution imagebuilder: ListImage BuildVersions imagebuilder: Pakete ListImage imagebuilder: ListImage PipelineImages imagebuilder: Rohrleitungen ListImage imagebuilder: Rezepte ListImage imagebuilder: ListImages imagebuilder: Aggregationen ListImage ScanFinding Imagebuilder: ListImage ScanFindings

Servicepräfix	Aktionen
	imagebuilder: Konfigurationen ListInfrastructure imagebuilder: ListLifecycle ExecutionResources imagebuilder: Ausführungen ListLifecycle imagebuilder: Richtlinien ListLifecycle imagebuilder: ListWaiting WorkflowSteps imagebuilder: Ausführungen ListWorkflow Imagebuilder: ListWorkflows Bildgenerator: ListWorkflow StepExecutions imagebuilder: Richtlinie PutComponent imagebuilder: PutContainer RecipePolicy imagebuilder: Richtlinie PutImage imagebuilder: PutImage RecipePolicy Image-Builder: SendWorkflow StepAction Image-Builder: StartImage PipelineExecution Image-Builder: StartResource StateUpdate imagebuilder: Konfiguration UpdateDistribution imagebuilder: Pipeline UpdateImage imagebuilder: Konfiguration UpdateInfrastructure

Servicepräfix	Aktionen
inspector	Inspektor: AddAttributes ToFindings
	Inspektor: CreateAssessment Ziel
	Inspektor: CreateAssessment Vorlage
	Inspektor: CreateExclusions Vorschau
	Inspektor: CreateResource Gruppe
	Inspektor: DeleteAssessment Lauf
	Inspektor: DeleteAssessment Ziel
	Inspektor: DeleteAssessment Vorlage
	Inspektor: DescribeAssessment Läuft
	Inspektor: DescribeAssessment Ziele
	Inspektor: DescribeAssessment Vorlagen
	Inspektor: DescribeCross AccountAccess Rolle
	Inspektor: DescribeExclusions
	Inspektor: DescribeFindings
	Inspektor: DescribeResource Gruppen
	Inspektor: DescribeRules Pakete
	Inspektor: GetAssessment Bericht
	Inspektor: GetExclusions Vorschau
	Inspektor: GetTelemetry Metadaten
	Inspektor: ListAssessment RunAgents
	Inspektor: ListAssessment Läuft

Servicepräfix	Aktionen
	Inspektor: ListAssessment Ziele
	Inspektor: ListAssessment Vorlagen
	Inspektor: ListEvent Abonnements
	Inspektor: ListExclusions
	Inspektor: ListFindings
	Inspektor: ListRules Pakete
	Inspektor: PreviewAgents
	Inspektor: RegisterCross AccountAccess Rolle
	Inspektor: RemoveAttributes FromFindings
	Inspektor: StartAssessment Lauf
	Inspektor: StopAssessment Lauf
	Inspektor: SubscribeTo Ereignis
	Inspektor: UnsubscribeFrom Ereignis
	Inspektor: UpdateAssessment Ziel

Servicepräfix	Aktionen
inspector2	Inspektor 2: AssociateMember Inspektor 2: BatchGet AccountStatus Inspektor 2: BatchGet CodeSnippet Inspektor 2: BatchGet FindingDetails Inspektor2: Information BatchGet FreeTrial Inspektor2:2 Status BatchGet MemberEc DeepInspection Inspektor2:2 Status BatchUpdate MemberEc DeepInspection Inspektor2: Bericht CancelFindings Inspektor2: Export CancelSbom Inspektor2: CreateCis ScanConfiguration Inspektor 2: CreateFilter Inspektor2: Bericht CreateFindings Inspektor2: Export CreateSbom Inspektor2: DeleteCis ScanConfiguration Inspektor 2: DeleteFilter inspector2: Konfiguration DescribeOrganization inspector2:Disable Inspektor2: DisableDelegated AdminAccount Inspektor 2: DisassociateMember inspector2:Enable Inspektor 2: EnableDelegated AdminAccount

Servicepräfix	Aktionen
	Inspektor 2: GetCis ScanReport Inspektor2: Einzelheiten GetCis ScanResult Inspektor2: GetConfiguration Inspektor 2: GetDelegated AdminAccount inspector2:2 Konfiguration GetEc DeepInspection inspector2: Schlüssel GetEncryption Inspektor2: GetFindings ReportStatus Inspektor 2: GetMember Inspektor2: Exportieren GetSbom inspector2: Berechtigungen ListAccount Inspektor2: ListCis ScanConfigurations Inspektor2: Schecks ListCis ScanResults AggregatedBy Inspektor2: ListCis ScanResults AggregatedBy TargetResource inspector2: scannt ListCis Inspektor2: ListCoverage Inspektor2: Statistik ListCoverage Inspektor2: ListDelegated AdminAccounts Inspektor 2: ListFilters inspector2: Aggregationen ListFinding Inspektor2: ListFindings Inspektor 2: ListMembers

Servicepräfix	Aktionen
	Inspektor2: Insgesamt ListUsage inspector2: Schlüssel ResetEncryption Inspektor2: SearchVulnerabilities Inspektor 2: SendCis SessionHealth Inspektor 2: SendCis SessionTelemetry Inspektor2: Sitzung StartCis inspector2: Sitzung StopCis Inspektor2: UpdateCis ScanConfiguration Inspektor 2: UpdateConfiguration inspector2:2 Konfiguration UpdateEc DeepInspection inspector2: Schlüssel UpdateEncryption Inspektor2: UpdateFilter inspector2: Konfiguration UpdateOrganization inspector2: Ec2-Konfiguration UpdateOrg DeepInspection

Servicepräfix	Aktionen
iot	iot: Übertragung AcceptCertificate
	iot: AddThing ToBilling Gruppe
	iot: AddThing ToThing Gruppe
	IoT: AssociateTargets WithJob
	IoT: AttachPolicy
	iot: AttachPrincipal Politik
	iot: AttachSecurity Profil
	iot: AttachThing Schulleiter
	iot: CancelAudit MitigationActions Aufgabe
	iot: CancelAudit Aufgabe
	iot: CancelCertificate Übertragung
	iot: CancelDetect MitigationActions Aufgabe
	IoT: CancelJob
	iot: CancelJob Ausführung
	iot: ClearDefault Autorisierer
	IoT: ConfirmTopic RuleDestination
	iot: CreateAudit Unterdrückung
	IoT: CreateAuthorizer
	iot: CreateBilling Gruppe
	IoT: CreateCertificate FromCsr
	iot: CreateCertificate Anbieter

Servicepräfix	Aktionen
	iot: CreateCustom Metrisch
	IoT: CreateDimension
	iot: CreateDomain Konfiguration
	iot: CreateDynamic ThingGroup
	iot: CreateFleet Metrisch
	IoT: CreateJob
	iot: CreateJob Vorlage
	IoT: CreateKeys AndCertificate
	iot: CreateMitigation Aktion
	iot:CreateOTAUpdate
	IoT: CreatePackage
	iot: CreatePackage Ausführung
	IoT: CreatePolicy
	iot: CreatePolicy Ausführung
	iot: CreateProvisioning Anspruch erheben
	iot: CreateProvisioning Vorlage
	IoT: CreateProvisioning TemplateVersion
	iot: CreateRole Aliasname
	iot: CreateScheduled Prüfung
	iot: CreateSecurity Profil
	iot: CreateStream

Servicepräfix	Aktionen
	IoT: CreateThing
	iot: CreateThing Gruppe
	iot: CreateThing Typ
	iot: CreateTopic Regel
	IoT: CreateTopic RuleDestination
	IoT: DeleteAccount AuditConfiguration
	iot: DeleteAudit Unterdrückung
	IoT: DeleteAuthorizer
	iot: DeleteBilling Gruppe
	iot:DeleteCACertificate
	IoT: DeleteCertificate
	iot: DeleteCertificate Anbieter
	iot: DeleteCustom Metrisch
	IoT: DeleteDimension
	iot: DeleteDomain Konfiguration
	iot: DeleteDynamic ThingGroup
	iot: DeleteFleet Metrisch
	IoT: DeleteJob
	iot: DeleteJob Ausführung
	iot: DeleteJob Vorlage
	iot: DeleteMitigation Aktion

Servicepräfix	Aktionen
	iot:DeleteOTAUpdate
	IoT: DeletePackage
	iot: DeletePackage Ausführung
	IoT: DeletePolicy
	iot: DeletePolicy Ausführung
	iot: DeleteProvisioning Vorlage
	IoT: DeleteProvisioning TemplateVersion
	IoT: DeleteRegistration Kode
	iot: DeleteRole Aliasname
	iot: DeleteScheduled Prüfung
	iot: DeleteSecurity Profil
	iot: DeleteStream
	IoT: DeleteThing
	iot: DeleteThing Gruppe
	iot: DeleteThing Typ
	iot: DeleteTopic Regel
	IoT: DeleteTopic RuleDestination
	IoT: V2 löschen LoggingLevel
	iot: Typ DeprecateThing
	iot: DescribeAccount AuditConfiguration
	iot: DescribeAudit Finden

Servicepräfix	Aktionen
	iot: DescribeAudit MitigationActions Aufgabe
	iot: DescribeAudit Unterdrückung
	iot: DescribeAudit Aufgabe
	IoT: DescribeAuthorizer
	iot: DescribeBilling Gruppe
	iot:DescribeCACertificate
	IoT: DescribeCertificate
	iot: DescribeCertificate Anbieter
	iot: DescribeCustom Metrisch
	iot: DescribeDefault Autorisierer
	iot: Aufgabe DescribeDetect MitigationActions
	IoT: DescribeDimension
	iot: DescribeDomain Konfiguration
	iot: DescribeEndpoint
	iot: DescribeEvent Konfigurationen
	iot: DescribeFleet Metrisch
	IoT: DescribeIndex
	IoT: DescribeJob
	iot: DescribeJob Ausführung
	iot: DescribeJob Vorlage
	IoT: DescribeManaged JobTemplate

Servicepräfix	Aktionen
	iot: DescribeMitigation Aktion
	iot: DescribeProvisioning Vorlage
	IoT: DescribeProvisioning TemplateVersion
	iot: DescribeRole Aliasname
	iot: DescribeScheduled Prüfung
	iot: DescribeSecurity Profil
	iot: DescribeStream
	IoT: DescribeThing
	iot: DescribeThing Gruppe
	IoT: DescribeThing RegistrationTask
	iot: DescribeThing Typ
	iot: DetachPolicy
	iot: DetachPrincipal Politik
	iot: DetachSecurity Profil
	iot: DetachThing Schulleiter
	iot: DisableTopic Regel
	iot: EnableTopic Regel
	iot: GetBehavior ModelTraining Zusammenfassungen
	iot: Aggregation GetBuckets
	IoT: GetCardinality
	iot: GetEffective Richtlinien

Servicepräfix	Aktionen
	iot: GetJob Dokument
	iot: GetLogging Optionen
	iot:GetOTAUpdate
	IoT: GetPackage
	iot: GetPackage Konfiguration
	iot: GetPackage Ausführung
	IoT: GetPercentiles
	IoT: GetPolicy
	iot: GetPolicy Ausführung
	iot: GetRegistration Kode
	IoT: GetStatistics
	iot: GetTopic Regel
	IoT: GetTopic RuleDestination
	IoT: GET V2 LoggingOptions
	iot: Verstöße ListActive
	iot: ListAttached Richtlinien
	iot: ListAudit Ergebnisse
	iot: ListAudit MitigationActions Hinrichtungen
	iot: Aufgaben ListAudit MitigationActions
	iot: ListAudit Unterdrückungen
	iot: Aufgaben ListAudit

Servicepräfix	Aktionen
	IoT: ListAuthorizers
	iot: ListBilling Gruppen
	iot:ListCACertificates
	iot: ListCertificate Anbieter
	IoT: ListCertificates
	iot: ListCertificates von CA
	iot: Metriken ListCustom
	iot: ListDetect MitigationActions Hinrichtungen
	iot: Aufgaben ListDetect MitigationActions
	IoT: ListDimensions
	iot: ListDomain Konfigurationen
	iot: ListFleet Metriken
	IoT: ListIndices
	iot: ListJob ExecutionsFor Job
	IoT: ListJob ExecutionsFor Sache
	IoT: ListJobs
	iot: ListJob Vorlagen
	IoT: ListManaged JobTemplates
	iot: ListMetric Werte
	iot: ListMitigation Aktionen
	iot:ListOTAUpdates

Servicepräfix	Aktionen
	iot: ListOutgoing Zertifikate
	iot: ListPackages
	iot: ListPackage Versionen
	iot: ListPolicies
	iot: ListPolicy Prinzipien
	iot: Versionen ListPolicy
	iot: ListPrincipal Richtlinien
	IoT: ListPrincipal Dinge
	iot: ListProvisioning Vorlagen
	IoT: ListProvisioning TemplateVersions
	IoT: ListRelated ResourcesFor AuditFinding
	iot: ListRole Aliase
	iot: Prüfungen ListScheduled
	iot: ListSecurity Profile
	iot: ListSecurity ProfilesFor Ziel
	IoT: ListStreams
	IoT: ListTargets ForPolicy
	iot: ListTargets ForSecurity Profil
	iot: ListThing Gruppen
	iot: ListThing GroupsFor Sache
	IoT: ListThing Schulleiter

Servicepräfix	Aktionen
	iot: Berichte ListThing RegistrationTask
	IoT: ListThing RegistrationTasks
	IoT: ListThings
	iot: ListThings InBilling Gruppe
	iot: ListThings InThing Gruppe
	iot: ListThing Typen
	IoT: ListTopic RuleDestinations
	iot: ListTopic Regeln
	IoT: Liste V2 LoggingLevels
	iot: Ereignisse ListViolation
	iot: PutVerification StateOn Verstoß
	iot:RegisterCACertificate
	IoT: RegisterCertificate
	iot: RegisterCertificate Ohne CA
	IoT: RegisterThing
	iot: RejectCertificate Übertragung
	iot: RemoveThing FromBilling Gruppe
	iot: RemoveThing FromThing Gruppe
	iot: ReplaceTopic Regel
	IoT: SearchIndex
	iot: SetDefault Autorisierer

Servicepräfix	Aktionen
	IoT: SetDefault PolicyVersion
	iot: SetLogging Optionen
	IoT: SETv2 LoggingLevel
	IoT: SETV2 LoggingOptions
	iot: Aufgabe StartAudit MitigationActions
	iot: StartDetect MitigationActions Aufgabe
	iot: StartOn DemandAudit Aufgabe
	IoT: StartThing RegistrationTask
	IoT: StopThing RegistrationTask
	IoT: TestAuthorization
	iot: TestInvoke Autorisierer
	IoT: TransferCertificate
	IoT: UpdateAccount AuditConfiguration
	iot: UpdateAudit Unterdrückung
	IoT: UpdateAuthorizer
	iot: UpdateBilling Gruppe
	iot:UpdateCACertificate
	IoT: UpdateCertificate
	iot: UpdateCertificate Anbieter
	iot: UpdateCustom Metrisch
	IoT: UpdateDimension

Servicepräfix	Aktionen
	iot: UpdateDomain Konfiguration
	iot: UpdateDynamic ThingGroup
	iot: UpdateEvent Konfigurationen
	iot: UpdateFleet Metrisch
	iot: UpdateIndexing Konfiguration
	iot: UpdateJob
	iot: UpdateMitigation Aktion
	IoT: UpdatePackage
	iot: UpdatePackage Konfiguration
	iot: UpdatePackage Ausführung
	iot: UpdateProvisioning Vorlage
	iot: UpdateRole Alias
	iot: UpdateScheduled Prüfung
	iot: UpdateSecurity Profil
	iot: UpdateStream
	IoT: UpdateThing
	iot: UpdateThing Gruppe
	IoT: UpdateThing GroupsFor Sache
	IoT: UpdateTopic RuleDestination
	IoT: ValidateSecurity ProfileBehaviors

Servicepräfix	Aktionen
iotanalytics	IoT-Analytik: Wiederaufbereitung CancelPipeline IoT-Analytik: CreateChannel IoT-Analytik: CreateDataset iotanalytics: Inhalt CreateDataset iotanalytics: CreateDatastore IoT-Analytik: CreatePipeline IoT-Analytik: DeleteChannel IoT-Analytik: DeleteDataset iotanalytics: Inhalt DeleteDataset iotanalytics: DeleteDatastore IoT-Analytik: DeletePipeline IoT-Analytik: DescribeChannel IoT-Analytik: DescribeDataset IoT-Analytik: DescribeDatastore iotanalytics: Optionen DescribeLogging iotanalytics: DescribePipeline iotanalytics: Inhalt GetDataset iotanalytics: ListChannels iotanalytics: Inhalt ListDataset iotanalytics: ListDatasets IoT-Analytik: ListDatastores

Servicepräfix	Aktionen
	IoT-Analytik: ListPipelines iotanalytics: Optionen PutLogging iotanalytics: Aktivität RunPipeline iotanalytics: Daten SampleChannel iotanalytics: Wiederaufbereitung StartPipeline IoT-Analytik: UpdateChannel IoT-Analytik: UpdateDataset IoT-Analytik: UpdateDatastore IoT-Analytik: UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor: Definition CreateSuite iotdeviceadvisor: DeleteSuite Definition iotdeviceadvisor: GetEndpoint iotdeviceadvisor: Definition GetSuite iotdeviceadvisor: GetSuite Ausführen iotdeviceadvisor: GetSuite RunReport iotdeviceadvisor: Definitionen ListSuite iotdeviceadvisor: ListSuite Läuft iotdeviceadvisor: StartSuite Ausführen iotdeviceadvisor: StopSuite Ausführen iotdeviceadvisor: UpdateSuite Definition

Servicepräfix	Aktionen
iotevents	iotevents: BatchAcknowledge Alarm iotevents: Detektor BatchDelete iotevents: Alarm BatchDisable iotevents: Alarm BatchEnable iotevents: Alarm BatchReset iotevents: Alarm BatchSnooze iotevents: Detektor BatchUpdate iotevents: Modell CreateAlarm iotevents: Modell CreateDetector iotevents: CreateInput iotevents: Modell DeleteAlarm iotevents: Modell DeleteDetector iotevents: DeleteInput ioereignisse: DescribeAlarm iotevents: Modell DescribeAlarm iotevents: DescribeDetector iotevents: Modell DescribeDetector iotevents: DescribeDetector ModelAnalysis ioereignisse: DescribeInput iotevents: Optionen DescribeLogging iotevents: Ergebnisse GetDetector ModelAnalysis

Servicepräfix	Aktionen
	iotevents: Modelle ListAlarm iotevents: ListAlarm ModelVersions ioereignisse: ListAlarms iotevents: Modelle ListDetector iotevents: ListDetector ModelVersions ioereignisse: ListDetectors iotevents: Routings ListInput ioevents: ListInputs iotevents: Optionen PutLogging iotevents: StartDetector ModelAnalysis iotevents: Modell UpdateAlarm iotevents: Modell UpdateDetector iotevents: UpdateInput
iotfleethub	iotfleethub: CreateApplication iotfleethub: DeleteApplication iotfleethub: DescribeApplication iotfleethub: ListApplications iotfleethub: UpdateApplication

Servicepräfix	Aktionen
iotsitewise	iot-siteseitig: AssociateAssets iotsitweise: AssociateTime SeriesTo AssetProperty iotsitweise: BatchAssociate ProjectAssets iotsitweise: BatchDisassociate ProjectAssets iotsitewise: Wert BatchGet AssetProperty iotsitewise: BatchGet AssetProperty ValueHistory iotsitewise: Wert BatchPut AssetProperty iotsitewise: Richtlinie CreateAccess iotsitewise: CreateAsset iotsitewise: Modell CreateAsset iotsitewise: Modell CreateAsset ModelComposite iotsitewise: CreateBulk ImportJob iotsitewise: CreateDashboard iotsitweise: CreateGateway iotsitweise: CreatePortal iotsitweise: CreateProject iotsitewise: Richtlinie DeleteAccess iotsitewise: DeleteAsset iotsitewise: Modell DeleteAsset iotsitewise: Modell DeleteAsset ModelComposite iotsitewise: DeleteDashboard

Servicepräfix	Aktionen
	iotsitweise: DeleteGateway iotsitweise: DeletePortal iotsitweise: DeleteProject iotsitewise: Serie DeleteTime iotsitewise: Richtlinie DescribeAccess iotsitewise: DescribeAsset iotsitewise: DescribeAsset CompositeModel iotsitewise: Modell DescribeAsset iotsitewise: Modell DescribeAsset ModelComposite iotsitewise: Eigenschaft DescribeAsset iotsitewise: DescribeBulk ImportJob iotsitweise: DescribeDashboard iotsitweise: DescribeDefault EncryptionConfiguration iotsitweise: DescribeGateway iotsitweise: DescribeGateway CapabilityConfiguration iotsitewise: Optionen DescribeLogging iotsitewise: DescribePortal iotsitewise: DescribeProject iotsitewise: Konfiguration DescribeStorage iotsitewise: Serie DescribeTime iotsitewise: DisassociateAssets

Servicepräfix	Aktionen
	iotsitweise: DisassociateTime SeriesFrom AssetProperty iotsitweise: ExecuteAction iotsitweise: ExecuteQuery iotsitewise: Richtlinien ListAccess iotsitewise: ListActions iotsitewise: Modelle ListAsset ModelComposite iotsitewise: ListAsset ModelProperties iotsitewise: Modelle ListAsset iotsitewise: Eigenschaften ListAsset iotsitewise: Beziehungen ListAsset iotsitewise: ListAssets iotsitewise: Vermögenswerte ListAssociated iotsitewise: ListBulk ImportJobs iotsitewise: Beziehungen ListComposition iotsitewise: ListDashboards iotsitewise: ListGateways iotsitewise: ListPortals iotsitewise: Vermögenswerte ListProject iotsitewise: ListProjects iotsitewise: Serie ListTime iotsitewise: PutDefault EncryptionConfiguration

Servicepräfix	Aktionen
	iotsitewise: Optionen PutLogging iotsitewise: Konfiguration PutStorage iotsitewise: Richtlinie UpdateAccess iotsitewise: UpdateAsset iotsitewise: Modell UpdateAsset iotsitewise: Modell UpdateAsset ModelComposite iotsitewise: Eigenschaft UpdateAsset iotsitewise: UpdateDashboard iotsitweise: UpdateGateway iotsitweise: UpdateGateway CapabilityConfiguration iotsitweise: UpdatePortal iotsitweise: UpdateProject

Servicepräfix	Aktionen
iottwinmaker	iottwinmaker: CancelMetadata TransferJob
	iottwinmaker: Typ CreateComponent
	iottwinmaker: CreateEntity
	iottwinmaker: CreateMetadata TransferJob
	iottwinmaker: CreateScene
	iottwinmaker: Job CreateSync
	iottwinmaker: CreateWorkspace
	iottwinmaker: Typ DeleteComponent
	iottwinmaker: DeleteEntity
	iottwinmaker: DeleteScene
	iottwinmaker: Job DeleteSync
	iottwinmaker: DeleteWorkspace
	iottwinmaker: ExecuteQuery
	iottwinmaker: GetMetadata TransferJob
	iottwinmaker: Plan GetPricing
	iottwinmaker: GetScene
	iottwinmaker: Job GetSync
	iottwinmaker: ListComponents
	iottwinmaker: Typen ListComponent
	iottwinmaker: ListEntities
	iottwinmaker: ListMetadata TransferJobs

Servicepräfix	Aktionen
	iottwinmaker: ListProperties
	iottwinmaker: ListScenes
	iottwinmaker: Stellenangebote ListSync
	iottwinmaker: Ressourcen ListSync
	iottwinmaker: ListWorkspaces
	iottwinmaker: Typ UpdateComponent
	iottwinmaker: UpdateEntity
	iottwinmaker: Plan UpdatePricing
	iottwinmaker: UpdateScene
	iottwinmaker: UpdateWorkspace

Servicepräfix	Aktionen
iotwireless	iot drahtlos: AssociateAws AccountWith PartnerAccount IoT drahtlos: AssociateMulticast GroupWith FuotaTask IoT drahtlos: AssociateWireless DeviceWith FuotaTask IoT drahtlos: AssociateWireless DeviceWith MulticastGroup iotwireless: Sache AssociateWireless DeviceWith iotwireless: Zertifikat AssociateWireless GatewayWith iotwireless: Sache AssociateWireless GatewayWith iotwireless: CancelMulticast GroupSession IoT drahtlos: CreateDestination iotwireless: Profil CreateDevice iotwireless: Aufgabe CreateFuota iotwireless: Gruppe CreateMulticast iotwireless: CreateNetwork AnalyzerConfiguration iotwireless: Profil CreateService iotwireless: Gerät CreateWireless iotwireless: Gateway CreateWireless Internet der Dinge drahtlos: CreateWireless GatewayTask iotwireless: Definition CreateWireless GatewayTask iotwireless: DeleteDestination iotwireless: Profil DeleteDevice iotwireless: Aufgabe DeleteFuota

Servicepräfix	Aktionen
	iotwireless: Gruppe DeleteMulticast iotwireless: DeleteNetwork AnalyzerConfiguration iotwireless: Nachrichten DeleteQueued iotwireless: Profil DeleteService iotwireless: Gerät DeleteWireless iotwireless: Aufgabe DeleteWireless DeviceImport iotwireless: Gateway DeleteWireless Internet der Dinge drahtlos: DeleteWireless GatewayTask iotwireless: Definition DeleteWireless GatewayTask iotwireless: Gerät DeregisterWireless iotwireless: DisassociateAws AccountFrom PartnerAccount IoT drahtlos: DisassociateMulticast GroupFrom FuotaTask IoT drahtlos: DisassociateWireless DeviceFrom FuotaTask IoT drahtlos: DisassociateWireless DeviceFrom MulticastGroup iotwireless: Sache DisassociateWireless DeviceFrom iotwireless: Zertifikat DisassociateWireless GatewayFrom iotwireless: Sache DisassociateWireless GatewayFrom iotwireless: GetDestination iotwireless: Profil GetDevice iotwireless: GetEvent ConfigurationBy ResourceTypes iotwireless: Aufgabe GetFuota

Servicepräfix	Aktionen
	iotwireless: GetLog LevelsBy ResourceTypes iotwireless: Konfiguration GetMetric iotwireless: GetMetrics iotwireless: Gruppe GetMulticast iotwireless: GetMulticast GroupSession iot drahtlos: GetNetwork AnalyzerConfiguration iotwireless: Konto GetPartner iotwireless: GetPosition iotwireless: Konfiguration GetPosition iotwireless: Schätzung GetPosition iotwireless: GetResource EventConfiguration iot drahtlos: GetResource LogLevel iotwireless: Position GetResource iotwireless: Endpunkt GetService iotwireless: Profil GetService iotwireless: Gerät GetWireless iotwireless: Aufgabe GetWireless DeviceImport iotwireless: GetWireless DeviceStatistics iotwireless: Gateway GetWireless Internet der Dinge drahtlos: GetWireless GatewayCertificate iotwireless: Informationen GetWireless GatewayFirmware

Servicepräfix	Aktionen
	iotwireless: GetWireless GatewayStatistics iot drahtlos: GetWireless GatewayTask iotwireless: Definition GetWireless GatewayTask iotwireless: ListDestinations iotwireless: Profile ListDevice iotwireless: Aufgabe ListDevices ForWireless DeviceImport iotwireless: Konfigurationen ListEvent iotwireless: Aufgaben ListFuota iotwireless: Gruppen ListMulticast iotwireless: ListMulticast GroupsBy FuotaTask IoT drahtlos: ListNetwork AnalyzerConfigurations iotwireless: Konten ListPartner iotwireless: Konfigurationen ListPosition iotwireless: Nachrichten ListQueued iotwireless: Profile ListService iotwireless: Aufgaben ListWireless DeviceImport iotwireless: Geräte ListWireless iotwireless: Gateways ListWireless iotwireless: Definitionen ListWireless GatewayTask iotwireless: Konfiguration PutPosition iotwireless: PutResource LogLevel

Servicepräfix	Aktionen
	iotwireless: Stufen ResetAll ResourceLog iotwireless: ResetResource LogLevel iotwireless: Gruppe SendData ToMulticast iotwireless: Gerät SendData ToWireless iotwireless: StartBulk AssociateWireless DeviceWith MulticastGroup IoT drahtlos: StartBulk DisassociateWireless DeviceFrom Multicast Group iotwireless: Aufgabe StartFuota iotwireless: StartMulticast GroupSession iot drahtlos: StartNetwork AnalyzerStream IoT drahtlos: StartSingle WirelessDevice ImportTask iotwireless: Aufgabe StartWireless DeviceImport iotwireless: Gerät TestWireless iotwireless: UpdateDestination iot drahtlos: UpdateEvent ConfigurationBy ResourceTypes iotwireless: Aufgabe UpdateFuota iotwireless: UpdateLog LevelsBy ResourceTypes iotwireless: Konfiguration UpdateMetric iotwireless: Gruppe UpdateMulticast iotwireless: UpdateNetwork AnalyzerConfiguration iotwireless: Konto UpdatePartner

Servicepräfix	Aktionen
	iotwireless: UpdatePosition IoT drahtlos: UpdateResource EventConfiguration iotwireless: Position UpdateResource iotwireless: Gerät UpdateWireless iotwireless: Aufgabe UpdateWireless DeviceImport iotwireless: Gateway UpdateWireless

Servicepräfix	Aktionen
ivs	ivs: Kanal BatchGet
	ivs: BatchGet StreamKey
	ivs: Widerruf BatchStart ViewerSession
	ivs: CreateChannel
	ivs: Konfiguration CreateEncoder
	ivs: Token CreateParticipant
	ivs: CreatePlayback RestrictionPolicy
	ivs: Konfiguration CreateRecording
	ivs: Konfiguration CreateStorage
	ivs: Schlüssel CreateStream
	ivs: DeleteChannel
	ivs: Konfiguration DeleteEncoder
	ivs: DeletePlayback KeyPair
	ivs: DeletePlayback RestrictionPolicy
	ivs: Konfiguration DeleteRecording
	ivs: Konfiguration DeleteStorage
	ivs: Schlüssel DeleteStream
	ivs: DisconnectParticipant
	ivs: GetChannel
	ivs: GetComposition
	ivs: Konfiguration GetEncoder

Servicepräfix	Aktionen
	ivs: GetParticipant
	ivs: GetPlayback KeyPair
	ivs: GetPlayback RestrictionPolicy
	ivs: Konfiguration GetRecording
	ivs: Konfiguration GetStorage
	ivs: GetStream
	ivs: Schlüssel GetStream
	ivs: Sitzung GetStream
	ivs: ImportPlayback KeyPair
	ivs: ListChannels
	ivs: ListCompositions
	ivs: Konfigurationen ListEncoder
	ivs: Ereignisse ListParticipant
	ivs: ListParticipants
	ivs: ListPlayback KeyPairs
	ivs: ListPlayback RestrictionPolicies
	ivs: Konfigurationen ListRecording
	ivs: Konfigurationen ListStorage
	ivs: Schlüssel ListStream
	ivs: ListStreams
	ivs: Sitzungen ListStream

Servicepräfix	Aktionen
	ivs: PutMetadata ivs: StartComposition ivs: StartViewer SessionRevocation ivs: StopComposition ivs: StopStream ivs: UpdateChannel ivs: UpdatePlayback RestrictionPolicy
ivschat	ivschat: Wertmarke CreateChat ivschat: Konfiguration CreateLogging ivschat: CreateRoom ivschat: Konfiguration DeleteLogging ivschat: DeleteMessage ivschat: DeleteRoom ivschat: DisconnectUser ivschat: Konfiguration GetLogging ivschat: GetRoom ivschat: Konfigurationen ListLogging ivschat: ListRooms ivschat: SendEvent ivschat: Konfiguration UpdateLogging ivschat: UpdateRoom

Servicepräfix	Aktionen
kafka	Kafka: BatchAssociate ScramSecret
	Kafka: BatchDisassociate ScramSecret
	Kafka: CreateCluster
	Kafka: V2 CreateCluster
	Kafka: CreateConfiguration
	Kafka: CreateReplicator
	kafka: Verbindung CreateVpc
	kafka: DeleteCluster
	kafka: Richtlinie DeleteCluster
	kafka: DeleteConfiguration
	Kafka: DeleteReplicator
	kafka: Verbindung DeleteVpc
	kafka: DescribeCluster
	kafka: Betrieb DescribeCluster
	kafka: Operation V2 DescribeCluster
	Kafka: V2 DescribeCluster
	Kafka: DescribeConfiguration
	kafka: Überarbeitung DescribeConfiguration
	kafka: Verbindung DescribeVpc
	kafka: Makler GetBootstrap
	kafka: Richtlinie GetCluster

Servicepräfix	Aktionen
	kafka: GetCompatible KafkaVersions
	Kafka: ListClient VpcConnections
	kafka: Operationen ListCluster
	kafka: Operationen V2 ListCluster
	Kafka: ListClusters
	Kafka: V2 ListClusters
	kafka: Überarbeitungen ListConfiguration
	Kafka: ListConfigurations
	kafka: Versionen ListKafka
	kafka: ListNodes
	Kafka: ListReplicators
	kafka: Geheimnisse ListScram
	kafka: Verbindungen ListVpc
	kafka: Richtlinie PutCluster
	kafka: RebootBroker
	Kafka: RejectClient VpcConnection
	kafka: Graf UpdateBroker
	kafka: Speicher UpdateBroker
	kafka: Typ UpdateBroker
	kafka: Konfiguration UpdateCluster
	kafka: UpdateCluster KafkaVersion

Servicepräfix	Aktionen
	Kafka: UpdateConfiguration Kafka: UpdateConnectivity Kafka: UpdateMonitoring kafka: Informationen UpdateReplication Kafka: UpdateSecurity Kafka: UpdateStorage
kafkaconnect	kafka verbinden: CreateConnector kafkaconnect: Erweiterung CreateCustom kafkaconnect: CreateWorker Konfiguration kafkaconnect: DeleteConnector kafkaconnect: Erweiterung DeleteCustom kafkaconnect: DeleteWorker Konfiguration kafkaconnect: DescribeConnector kafkaconnect: Erweiterung DescribeCustom kafkaconnect: DescribeWorker Konfiguration kafkaconnect: ListConnectors kafkaconnect: Erweiterungen ListCustom kafkaconnect: ListWorker Konfigurationen kafkaconnect: UpdateConnector

Servicepräfix	Aktionen
kendra	Kendra: AssociateEntities ToExperience
	Kendra: AssociatePersonas ToEntities
	kendra: Dokument BatchDelete
	Kendra: Satz BatchDelete FeaturedResults
	Kendra: BatchGet DocumentStatus
	kendra: Dokument BatchPut
	kendra: Vorschläge ClearQuery
	Kendra: CreateAccess ControlConfiguration
	Kendra: Quelle CreateData
	Kendra: CreateExperience
	Kendra: CreateFaq
	Kendra: CreateFeatured ResultsSet
	Kendra: CreateIndex
	Kendra: Liste CreateQuery SuggestionsBlock
	Kendra: CreateThesaurus
	Kendra: Quelle DeleteData
	Kendra: DeleteExperience
	Kendra: DeleteFaq
	Kendra: DeleteIndex
	Kendra: Kartierung DeletePrincipal
	Kendra: Liste DeleteQuery SuggestionsBlock

Servicepräfix	Aktionen
	Kendra: DeleteThesaurus
	Kendra: DescribeAccess ControlConfiguration
	Kendra: Quelle DescribeData
	Kendra: DescribeExperience
	Kendra: DescribeFaq
	Kendra: DescribeFeatured ResultsSet
	Kendra: DescribeIndex
	Kendra: Kartierung DescribePrincipal
	Kendra: Liste DescribeQuery SuggestionsBlock
	Kendra: DescribeQuery SuggestionsConfig
	Kendra: DescribeThesaurus
	Kendra: DisassociateEntities FromExperience
	Kendra: DisassociatePersonas FromEntities
	Kendra: Vorschläge GetQuery
	Kendra: GetSnapshots
	Kendra: ListAccess ControlConfigurations
	Kendra: Quellen ListData
	Kendra: Jobs ListData SourceSync
	Kendra: Personas ListEntity
	kendra: Entitäten ListExperience
	Kendra: ListExperiences

Servicepräfix	Aktionen
	Kendra: ListFaqs
	Kendra: ListFeatured ResultsSets
	Kendra: ListGroups OlderThan OrderingId
	Kendra: ListIndices
	kendra: Listen ListQuery SuggestionsBlock
	Kendra: ListThesauri
	Kendra: Kartierung PutPrincipal
	kendra:Query
	kendra:Retrieve
	Kendra: Job StartData SourceSync
	Kendra: Job StopData SourceSync
	Kendra: SubmitFeedback
	Kendra: Quelle UpdateData
	Kendra: UpdateExperience
	Kendra: UpdateFeatured ResultsSet
	Kendra: UpdateIndex
	Kendra: Liste UpdateQuery SuggestionsBlock
	Kendra: UpdateQuery SuggestionsConfig
	Kendra: UpdateThesaurus

Servicepräfix	Aktionen
kinesis	Kinese: CreateStream Kinese: DecreaseStream RetentionPeriod Kinese: DeleteStream Kinesis: Verbraucher DeregisterStream Kinesis: DescribeLimits Kinese: DescribeStream Kinesis: Verbraucher DescribeStream Kinesis: Zusammenfassung DescribeStream Kinesis: Überwachung DisableEnhanced Kinesis: Überwachung EnableEnhanced Kinese: IncreaseStream RetentionPeriod Kinese: ListShards Kinesis: Verbraucher ListStream Kinesis: ListStreams Kinese: MergeShards Kinesis: Verbraucher RegisterStream Kinesis: SplitShard Kinesis: Verschlüsselung StartStream Kinesis: Verschlüsselung StopStream Kinesis: Zählen UpdateShard Kinesis: Modus UpdateStream

Servicepräfix	Aktionen
kinesisanalytics	Kinesis-Analytik: AddApplication CloudWatch LoggingOption
	Kinesisanalytics: Eingabe AddApplication
	kinesisanalytics: Konfiguration AddApplication InputProcessing
	kinesisanalytics: Ausgabe AddApplication
	kinesisanalytics: Quelle AddApplication ReferenceData
	Kinesisanalytics: AddApplication VpcConfiguration
	Kinesis-Analytik: CreateApplication
	Kinesis-Analytik: CreateApplication PresignedUrl
	Kinesisanalytics: Schnappschuss CreateApplication
	Kinesisanalytics: DeleteApplication
	Kinesis-Analytik: DeleteApplication CloudWatch LoggingOption
	kinesisanalytics: Konfiguration DeleteApplication InputProcessing
	kinesisanalytics: Ausgabe DeleteApplication
	kinesisanalytics: Quelle DeleteApplication ReferenceData
	kinesisanalytics: Schnappschuss DeleteApplication
	Kinesisanalytics: DeleteApplication VpcConfiguration
	Kinesis-Analytik: DescribeApplication
	Kinesisanalytics: Schnappschuss DescribeApplication
	kinesisanalytics: Version DescribeApplication
	kinesisanalytics: Schema DiscoverInput
	Kinesis-Analytik: ListApplications

Servicepräfix	Aktionen
	Kinesisanalytics: Schnappschüsse ListApplication kinesisanalyticsListApplication: Versionen kinesisanalytics: RollbackApplication Kinesis-Analytik: StartApplication Kinesis-Analytik: StopApplication Kinesis-Analytik: UpdateApplication Kinesis-Analytik: UpdateApplication MaintenanceConfiguration

Servicepräfix	Aktionen
kms	kms: Löschung CancelKey km: ConnectCustom KeyStore km: CreateAlias km: CreateCustom KeyStore km: CreateGrant km: CreateKey kms:Decrypt km: DeleteAlias km: DeleteCustom KeyStore km: DeleteImported KeyMaterial km: DescribeCustom KeyStores km: DescribeKey km: DisableKey km: DisableKey Drehung km: DisconnectCustom KeyStore km: EnableKey km: EnableKey Drehung kms:Encrypt km: GenerateData Schlüssel km: GenerateData KeyPair km: GenerateData KeyPair WithoutPlaintext

Servicepräfix	Aktionen
	km: GenerateData KeyWithout Klartext km: GenerateMac km: GenerateRandom kms: GetKey Richtlinie km: GetKey RotationStatus km: GetParameters ForImport km: GetPublic Schlüssel km: ImportKey Material km: ListAliases km: ListGrants km: ListKey Richtlinien km: ListKeys km: ListRetirable Zuschüsse km: ReplicateKey km: RetireGrant km: RevokeGrant km: ScheduleKey Löschung kms:Sign km: UpdateAlias km: UpdateCustom KeyStore km: UpdateKey Beschreibung

Servicepräfix	Aktionen
	km: UpdatePrimary Region kms:Verify km: VerifyMac

Servicepräfix	Aktionen
Lambda	Lambda: AddLayer VersionPermission
	Lambda: AddLayer VersionPermission
	Lambda: AddPermission
	Lambda: AddPermission
	Lambda: AddPermission
	Lambda: CreateAlias
	Lambda: CreateAlias
	Lambda: CreateCode SigningConfig
	Lambda: CreateEvent SourceMapping
	Lambda: CreateEvent SourceMapping
	Lambda: CreateFunction
	Lambda: CreateFunction
	Lambda: CreateFunction UrlConfig
	Lambda: DeleteAlias
	Lambda: DeleteAlias
	Lambda: DeleteCode SigningConfig
	Lambda: DeleteEvent SourceMapping
	Lambda: DeleteEvent SourceMapping
	Lambda: DeleteFunction
	Lambda: DeleteFunction
	Lambda: Config DeleteFunction CodeSigning

Servicepräfix	Aktionen
	Lambda: Parallelität DeleteFunction
	Lambda: Parallelität DeleteFunction
	Lambda: Config DeleteFunction EventInvoke
	Lambda: DeleteFunction UrlConfig
	Lambda: Ausführung DeleteLayer
	Lambda: Ausführung DeleteLayer
	Lambda: DeleteProvisioned ConcurrencyConfig
	lambda: Einstellungen GetAccount
	lambda: Einstellungen GetAccount
	Lambda: GetAlias
	Lambda: GetAlias
	Lambda: GetCode SigningConfig
	Lambda: GetEvent SourceMapping
	Lambda: GetEvent SourceMapping
	Lambda: GetFunction
	Lambda: GetFunction
	Lambda: GetFunction
	Lambda: Config GetFunction CodeSigning
	Lambda: Parallelität GetFunction
	Lambda: Konfiguration GetFunction
	lambda: Konfiguration GetFunction

Servicepräfix	Aktionen
	lambda: Konfiguration GetFunction
	Lambda: Config GetFunction EventInvoke
	Lambda: GetFunction UrlConfig
	Lambda: Ausführung GetLayer
	Lambda: Ausführung GetLayer
	Lambda: Ausführung GetLayer
	Lambda: Ausführung GetLayer
	Lambda: GetLayer VersionPolicy
	Lambda: GetLayer VersionPolicy
	Lambda: GetPolicy
	Lambda: GetPolicy
	Lambda: GetPolicy
	Lambda: GetProvisioned ConcurrencyConfig
	Lambda: GetRuntime ManagementConfig
	Lambda: ListAliases
	Lambda: ListAliases
	Lambda: ListCode SigningConfigs
	Lambda: ListEvent SourceMappings
	Lambda: ListEvent SourceMappings
	lambda: Konfigurationen ListFunction EventInvoke
	Lambda: ListFunctions

Servicepräfix	Aktionen
	Lambda: ListFunctions
	Lambda: ListFunctions ByCode SigningConfig
	Lambda: ListFunction UrlConfigs
	Lambda: ListLayers
	Lambda: ListLayers
	lambda: Versionen ListLayer
	lambda: Versionen ListLayer
	Lambda: ListProvisioned ConcurrencyConfigs
	Lambda: ListVersions ByFunction
	Lambda: ListVersions ByFunction
	Lambda: Ausführung PublishLayer
	Lambda: Ausführung PublishLayer
	Lambda: PublishVersion
	Lambda: PublishVersion
	Lambda: Config PutFunction CodeSigning
	Lambda: Parallelität PutFunction
	Lambda: Parallelität PutFunction
	Lambda: Config PutFunction EventInvoke
	Lambda: PutProvisioned ConcurrencyConfig
	Lambda: PutRuntime ManagementConfig
	Lambda: RemoveLayer VersionPermission

Servicepräfix	Aktionen
	Lambda: RemoveLayer VersionPermission
	Lambda: RemovePermission
	Lambda: RemovePermission
	Lambda: RemovePermission
	Lambda: UpdateAlias
	Lambda: UpdateAlias
	Lambda: UpdateCode SigningConfig
	Lambda: UpdateEvent SourceMapping
	Lambda: UpdateEvent SourceMapping
	Lambda: Kode UpdateFunction
	Lambda: Kode UpdateFunction
	Lambda: Kode UpdateFunction
	lambda: Konfiguration UpdateFunction
	lambda: Konfiguration UpdateFunction
	lambda: Konfiguration UpdateFunction
	Lambda: Config UpdateFunction EventInvoke
	Lambda: UpdateFunction UrlConfig

Servicepräfix	Aktionen
lex	Alex: Artikel BatchCreate CustomVocabulary lex: BatchDelete CustomVocabulary Artikel lex: BatchUpdate CustomVocabulary Artikel lex: BuildBot Gebietsschema lex: Aliasname CreateBot lex: CreateBot Ausführung lex: CreateExport lex: CreateIntent Ausführung lex: CreateResource Richtlinie Lex: CreateSlot TypeVersion lex: CreateTest SetDiscrepancy Bericht Lex: CreateUpload URL lex: DeleteBot Lex: DeleteBot ChannelAssociation Lex: DeleteExport Lex: DeleteImport lex: DeleteIntent Ausführung lex: DeleteResource Richtlinie Lex: DeleteSlot TypeVersion Alex: DeleteTest Satz lex: DeleteUtterances

Servicepräfix	Aktionen
	Lex: DescribeBot Deckname Lex: DescribeBot Empfehlung Lex: DescribeBot ResourceGeneration lex: DescribeBot Ausführung lex: DescribeCustom VocabularyMetadata Lex: DescribeExport Lex: DescribeImport Lex: DescribeResource Politik lex: DescribeTest Ausführung lex: DescribeTest Satz lex: DescribeTest SetDiscrepancy Bericht Lex: DescribeTest SetGeneration lex: GenerateBot Elemente lex: GetBot Lex: GetBot Deckname lex: GetBot Aliase Lex: GetBot ChannelAssociation Lex: GetBot ChannelAssociations Lex: GetBots lex: GetBot Versionen lex: GetBuiltin Absicht

Servicepräfix	Aktionen
	Alex: GetBuiltin Absichten Lex: GetBuiltin SlotTypes Lex: GetExport Lex: GetImport Lex: GetIntent Lex: GetIntents lex: GetIntent Versionen lex: GetMigration Lex: GetMigrations lex: GetSlot Typ lex: GetSlot Typen lex: GetSlot TypeVersions lex: GetTest ExecutionArtifacts URL lex: GetUtterances Ansehen lex: ListBot Aliase Lex: Empfehlungen ListBot Lex: ListBot ResourceGenerations Lex: ListBots lex: ListBot Versionen lex: ListBuilt InIntents lex: ListBuilt InSlot Typen

Servicepräfix	Aktionen
	lex: ListCustom VocabularyItems Lex: ListExports Lex: ListImports lex: ListIntent Metriken lex: ListIntent Pfade Lex: ListRecommended Absichten Lex: ListSession AnalyticsData lex: ListSession Metriken lex: ListTest ExecutionResult Artikel Lex: ListTest Hinrichtungen lex: Sätze ListTest lex: PutBot Lex: PutBot Deckname Lex: PutIntent lex: PutSlot Typ lex: SearchAssociated Transkripte lex: Empfehlung StartBot Lex: StartImport Lex: StartMigration lex: StartTest Hinrichtung Lex: StartTest SetGeneration

Servicepräfix	Aktionen
	Lex: StopBot Empfehlung Lex: UpdateBot Alias Lex: UpdateBot Empfehlung Lex: UpdateExport Lex: UpdateResource Politik
license-manager-linux-subscriptions	license-manager-linux-subscriptions: Einstellungen GetService license-manager-linux-abonnements: ListLinux SubscriptionInstances license-manager-linux-subscriptions: Abonnements ListLinux license-manager-linux-subscriptions: Einstellungen UpdateService

Servicepräfix	Aktionen
lightsail	Lightsail: Ip AllocateStatic Lichtsegel: AttachCertificate ToDistribution Lichtsegel: AttachDisk Lichtsegel: Balancer AttachInstances ToLoad lightsail: Zertifikat AttachLoad BalancerTls Lichtsegel: Ip AttachStatic Lichtsegel: CloseInstance PublicPorts Lichtsegel: CopySnapshot Lichtsegel: CreateBucket Lichtsegel: CreateBucket AccessKey Lichtsegel: CreateCertificate Lichtsegel: CreateCloud FormationStack Lichtsegel: Methode CreateContact Lightsail: Service CreateContainer Lichtsegel: CreateContainer ServiceDeployment lightsail: Einloggen CreateContainer ServiceRegistry Lichtsegel: CreateDisk Lichtsegel: CreateDisk FromSnapshot Lichtsegel: Schnappschuss CreateDisk Lichtsegel: CreateDistribution Lichtsegel: CreateDomain

Servicepräfix	Aktionen
	Lightsail: GUI-Details erstellen SessionAccess Lichtsegel: CreateInstances Lichtsegel: CreateInstances FromSnapshot Lichtsegel: Schnappschuss CreateInstance Lichtsegel: Paar CreateKey Lichtsegel: Balancer CreateLoad lightsail: Zertifikat CreateLoad BalancerTls lightsail: Datenbank CreateRelational lightsail: Schnappschuss CreateRelational DatabaseFrom Lichtsegel: CreateRelational DatabaseSnapshot Lichtsegel: DeleteAlarm Lichtsegel: Schnappschuss DeleteAuto Lichtsegel: DeleteBucket Lichtsegel: DeleteBucket AccessKey Lichtsegel: DeleteCertificate Lichtsegel: Methode DeleteContact Lichtsegel: Bild DeleteContainer Lichtsegel: Service DeleteContainer Lichtsegel: DeleteDisk Lichtsegel: Schnappschuss DeleteDisk Lichtsegel: DeleteDistribution

Servicepräfix	Aktionen
	Lichtsegel: DeleteDomain Lichtsegel: Eintrag DeleteDomain Lichtsegel: DeleteInstance Lichtsegel: Schnappschuss DeleteInstance Lichtsegel: Paar DeleteKey Lichtsegel: DeleteKnown HostKeys Lichtsegel: Balancer DeleteLoad lightsail: Zertifikat DeleteLoad BalancerTls lightsail: Datenbank DeleteRelational Lichtsegel: DeleteRelational DatabaseSnapshot Lichtsegel: DetachCertificate FromDistribution Lichtsegel: DetachDisk Lichtsegel: Balancer DetachInstances FromLoad Lichtsegel: Ip DetachStatic Lichtsegel: An DisableAdd Lichtsegel: DownloadDefault KeyPair Lichtsegel: An EnableAdd Lichtsegel: ExportSnapshot Lichtsegel: Namen GetActive Lichtsegel: GetAlarms Lichtsegel: Schnappschüsse GetAuto

Servicepräfix	Aktionen
	Lichtsegel: GetBlueprints Lichtsegel: GetBucket AccessKeys Lightsail: Bündel GetBucket Lichtsegel: GetBucket MetricData Lichtsegel: GetBuckets Lichtsegel: GetBundles Lichtsegel: GetCertificates Lightsail: Rekorde GetCloud FormationStack Lightsail: Methoden GetContact lightsail: API-Metadata GetContainer lightsailGetContainer: Bilder Lichtsegel: Protokoll GetContainer Lichtsegel: GetContainer ServiceDeployments Lichtsegel: Daten GetContainer ServiceMetric Lichtsegel: GetContainer ServicePowers Lightsail: Dienstleistungen GetContainer lightsail: Schätzung GetCost Lichtsegel: GetDisk Lichtsegel: GetDisks Lichtsegel: Schnappschuss GetDisk Lichtsegel: Schnappschüsse GetDisk

Servicepräfix	Aktionen
	lightsail: Pakete GetDistribution Lightsail: Zurücksetzen GetDistribution LatestCache Lichtsegel: GetDistribution MetricData Lichtsegel: GetDistributions Lichtsegel: GetDomain Lichtsegel: GetExport SnapshotRecords Lichtsegel: GetInstance Lichtsegel: GetInstance MetricData Lichtsegel: GetInstance PortStates Lichtsegel: GetInstances Lichtsegel: Schnappschuss GetInstance Lichtsegel: Schnappschüsse GetInstance Lightsail: Bundesstaat GetInstance Lichtsegel: Paar GetKey Lichtsegel: Paare GetKey Lichtsegel: Balancer GetLoad lightsail: Daten GetLoad BalancerMetric lightsail: Balancer GetLoad lightsail: Zertifikate GetLoad BalancerTls lightsail: Richtlinien GetLoad BalancerTls Lightsail: GetOperation

Servicepräfix	Aktionen
	Lichtsegel: GetOperations Lichtsegel: GetOperations ForResource Lichtsegel: GetRegions Lightsail: Datenbank GetRelational Lichtsegel: GetRelational DatabaseBlueprints Lichtsegel: GetRelational DatabaseBundles Lichtsegel: GetRelational DatabaseEvents Lightsail: Ereignisse GetRelational DatabaseLog lightsail: Streams GetRelational DatabaseLog Lichtsegel: GetRelational DatabaseMaster UserPassword Lichtsegel: Daten GetRelational DatabaseMetric Lichtsegel: GetRelational DatabaseParameters lightsail: Datenbanken GetRelational Lightsail: GetRelational DatabaseSnapshot Lichtsegel: GetRelational DatabaseSnapshots Lichtsegel: Geschichte GetSetup Lichtsegel: Ip GetStatic Lichtsegel: Ips GetStatic Lichtsegel: Paar ImportKey Lichtsegel: Peered IsVpc Lichtsegel: OpenInstance PublicPorts

Servicepräfix	Aktionen
	Lichtsegel: PeerVpc Lichtsegel: PutAlarm Lichtsegel: PutInstance PublicPorts Lichtsegel: RebootInstance Lightsail: Datenbank RebootRelational Lightsail: Bild RegisterContainer Lichtsegel: Ip ReleaseStatic Lichtsegel: Cache ResetDistribution Lichtsegel: SendContact MethodVerification Lichtsegel: SetIp AddressType Lichtsegel: Eimer SetResource AccessFor Lichtsegel: Https SetupInstance lightsail:StartGUISession Lichtsegel: StartInstance Lightsail: Datenbank StartRelational lightsail:StopGUISession Lichtsegel: StopInstance Lightsail: Datenbank StopRelational Lichtsegel: TestAlarm Lichtsegel: UnpeerVpc Lichtsegel: UpdateBucket

Servicepräfix	Aktionen
	Lichtsegel: Paket UpdateBucket lightsail: Service UpdateContainer Lichtsegel: UpdateDistribution Lichtsegel: Paket UpdateDistribution lightsail: Eintrag UpdateDomain Lichtsegel: UpdateInstance MetadataOptions Lichtsegel: UpdateLoad BalancerAttribute Lightsail: Datenbank UpdateRelational Lichtsegel: UpdateRelational DatabaseParameters

Servicepräfix	Aktionen
Protokolle	Logs: Schlüssel AssociateKms
	Protokolle: CancelExport Aufgabe
	Protokolle: CreateExport Aufgabe
	Protokolle: CreateLog AnomalyDetector
	Protokolle: CreateLog Gruppe
	Protokolle: CreateLog Stream
	Logs: DeleteData ProtectionPolicy
	Protokolle: DeleteDelivery
	protokolle: DeleteDelivery Ziel
	Protokolle: DeleteDelivery DestinationPolicy
	Logs: DeleteDelivery Quelle
	Protokolle: DeleteDestination
	Protokolle: DeleteLog Gruppe
	Protokolle: DeleteLog Stream
	Protokolle: DeleteMetric Filter
	Protokolle: DeleteQuery Definition
	Protokolle: DeleteResource Richtlinie
	Protokolle: DeleteRetention Richtlinie
	Protokolle: DeleteSubscription Filter
	Protokolle: DescribeAccount Richtlinien
	Protokolle: DescribeDeliveries

Servicepräfix	Aktionen
	Logs: DescribeDelivery Ziele
	Logs: DescribeDelivery Quellen
	Logs: DescribeDestinations
	Protokolle: DescribeExport Aufgaben
	Protokolle: DescribeLog Gruppen
	Protokolle: DescribeLog Streams
	Logs: DescribeMetric Filter
	Logs: DescribeQueries
	Protokolle: DescribeQuery Definitionen
	Protokolle: DescribeResource Richtlinien
	Protokolle: DescribeSubscription Filter
	Protokolle: DisassociateKms Schlüssel
	Protokolle: GetData ProtectionPolicy
	Protokolle: GetDelivery
	protokolle: GetDelivery Ziel
	Protokolle: GetDelivery DestinationPolicy
	Logs: GetDelivery Quelle
	Logs: GetLog GroupFields
	Logs: GetLog Rekord
	Protokolle: GetQuery Ergebnisse
	Protokolle: ListAnomalies

Servicepräfix	Aktionen
	Protokolle: ListLog AnomalyDetectors
	Protokolle: PutData ProtectionPolicy
	protokolle: PutDelivery Ziel
	Protokolle: PutDelivery DestinationPolicy
	Logs: PutDelivery Quelle
	Protokolle: PutDestination
	Protokolle: PutDestination Richtlinie
	Protokolle: PutMetric Filter
	Protokolle: PutQuery Definition
	Protokolle: PutResource Richtlinie
	Protokolle: PutRetention Richtlinie
	Protokolle: PutSubscription Filter
	Protokolle: StartLive Schwanz
	Logs: StartQuery
	Protokolle: StopQuery
	Protokolle: TestMetric Filter

Servicepräfix	Aktionen
lookoutequipment	Ausrüstung aussuchen: CreateDataset Lookoutequipment: Scheduler CreateInference Ausrüstung aussuchen: CreateLabel Lookoutequipment: Gruppe CreateLabel Ausrüstung ausfindig machen: CreateModel Ausrüstung zum Aussuchen: DeleteDataset Lookoutequipment: Scheduler DeleteInference Ausrüstung aussuchen: DeleteLabel Lookoutequipment: Gruppe DeleteLabel Ausrüstung ausfindig machen: DeleteModel Lookoutequipment: Richtlinie DeleteResource Lookoutequipment: DeleteRetraining Scheduler Ausrüstung aussuchen: DescribeData IngestionJob Ausrüstung zum Aussuchen: DescribeDataset Lookoutequipment: Scheduler DescribeInference lookoutequipment:Describelabel lookoutequipment: DescribeLabel Gruppe Ausrüstung ausfindig machen: DescribeModel Aussichtsausrüstung: Version DescribeModel lookoutequipment: DescribeResource Richtlinie Lookoutequipment: DescribeRetraining Scheduler

Servicepräfix	Aktionen
	Ausrüstung aussuchen: ImportDataset Aussichtsausrüstung: Version ImportModel Lookout-Ausrüstung: ListData IngestionJobs Ausrüstung zum Aussuchen: ListDatasets Lookoutequipment: Veranstaltungen ListInference Lookoutequipment: ListInference Hinrichtungen lookoutequipment: ListInference Planer Lookoutequipment: ListLabel Gruppen Ausrüstung ausfindig machen: ListLabels Ausrüstung zum Aussuchen: ListModels Lookoutequipment: Versionen ListModel lookoutequipment: ListRetraining Scheduler lookoutequipment: ListSensor Statistiken lookoutequipment: PutResource Richtlinie Lookout-Equipment: StartData IngestionJob Lookoutequipment: Scheduler StartInference Lookoutequipment: StartRetraining Scheduler Lookoutequipment: StopInference Scheduler Lookoutequipment: StopRetraining Scheduler Ausrüstung aussuchen: UpdateActive ModelVersion Lookoutequipment: Scheduler UpdateInference

Servicepräfix	Aktionen
	lookoutequipment: UpdateLabel Gruppe Ausrüstung ausfindig machen: UpdateModel Lookoutequipment: Scheduler UpdateRetraining

Servicepräfix	Aktionen
lookoutmetrics	lookoutmetrics: ActivateAnomaly Detektor
	Lookout-Metriken: BackTest AnomalyDetector
	Lookout-Metriken: CreateAlert
	lookoutmetrics: Detektor CreateAnomaly
	lookoutmetrics: Eingestellt CreateMetric
	lookoutmetrics: Detektor DeactivateAnomaly
	Lookout-Metriken: DeleteAlert
	lookoutmetrics: Detektor DeleteAnomaly
	Lookout-Metriken: DescribeAlert
	Lookout-Metriken: DescribeAnomaly DetectionExecutions
	lookoutmetrics: Detektor DescribeAnomaly
	lookoutmetrics: Eingestellt DescribeMetric
	Lookout-Metriken: DetectMetric SetConfig
	lookoutmetrics: Gruppe GetAnomaly
	Lookoutmetrics: GetData QualityMetrics
	Lookout-Metriken: GetFeedback
	lookoutmetrics: Daten GetSample
	Lookout-Metriken: ListAlerts
	lookoutmetrics: Detektoren ListAnomaly
	lookoutmetrics: Metriken ListAnomaly GroupRelated
	Lookout-Metriken: ListAnomaly GroupSummaries

Servicepräfix	Aktionen
	lookoutmetrics: Serie ListAnomaly GroupTime
	lookoutmetrics: Sätze ListMetric
	Lookout-Metriken: PutFeedback
	Lookout-Metriken: UpdateAlert
	lookoutmetrics: Detektor UpdateAnomaly
	lookoutmetrics: Eingestellt UpdateMetric

Servicepräfix	Aktionen
lookoutvision	lookoutvision: CreateDataset Lookoutvision: CreateModel Lookoutvision: CreateProject Lookoutvision: DeleteDataset Lookoutvision: DeleteModel Lookoutvision: DeleteProject Lookoutvision: DescribeDataset Lookoutvision: DescribeModel Lookoutvision: DescribeModel PackagingJob Lookoutvision: DescribeProject Lookoutvision: DetectAnomalies lookoutvision: Einträge ListDataset lookoutvision: ListModel PackagingJobs Lookoutvision: ListModels Lookoutvision: ListProjects Lookoutvision: StartModel Lookoutvision: StartModel PackagingJob Lookoutvision: StopModel lookoutvision: Einträge UpdateDataset

Servicepräfix	Aktionen
m2	m2: CancelBatch JobExecution
	m2: CreateApplication
	m2: CreateData SetImport Aufgabe
	m2: CreateDeployment
	m2: CreateEnvironment
	m2: DeleteApplication
	m2: DeleteApplication FromEnvironment
	m2: DeleteEnvironment
	m2: GetApplication
	m2: GetApplication Ausführung
	m2: GetBatch JobExecution
	m2: GetData SetDetails
	m2: GetData SetImport Aufgabe
	m2: GetDeployment
	m2: GetEnvironment
	m2: GetSigned BluinsightsUrl
	m2: ListApplications
	m2: ListApplication Versionen
	m2: ListBatch JobDefinitions
	m2: ListBatch JobExecutions
	m2: ListBatch JobRestart Punkte

Servicepräfix	Aktionen
	m2: ListData SetImport Geschichte m2: ListData Sets m2: ListDeployments m2: ListEngine Versionen m2: ListEnvironments m2: StartApplication m2: StartBatch Job m2: StopApplication m2: UpdateApplication m2: UpdateEnvironment

Servicepräfix	Aktionen
managedblockchain	verwaltete Blockchain: CreateAccessor verwaltete Blockchain: CreateMember verwaltete Blockchain: CreateNetwork verwaltete Blockchain: CreateNode verwaltete Blockchain: CreateProposal verwaltete Blockchain: DeleteAccessor verwaltete Blockchain: DeleteMember verwaltete Blockchain: DeleteNode verwaltete Blockchain: GetAccessor verwaltete Blockchain: GetMember verwaltete Blockchain: GetNetwork verwaltete Blockchain: GetNode verwaltete Blockchain: GetProposal verwaltete Blockchain: InvokeRpc PolygonMainnet verwaltete Blockchain: Testnet InvokeRpc PolygonMumbai verwaltete Blockchain: ListAccessors verwaltete Blockchain: ListInvitations verwaltete Blockchain: ListMembers verwaltete Blockchain: ListNetworks verwaltete Blockchain: ListNodes verwaltete Blockchain: ListProposals

Servicepräfix	Aktionen
	managedblockchain: Stimmen ListProposal verwaltete Blockchain: RejectInvitation verwaltete Blockchain: UpdateMember verwaltete Blockchain: UpdateNode managedblockchain: Vorschlag VoteOn

Servicepräfix	Aktionen
mediaconnect	mediaconnect: Ausgänge AddBridge
	mediaconnect: Quellen AddBridge
	mediaconnect: AddFlow MediaStreams
	mediaconnect: Ausgänge AddFlow
	mediaconnect: Quellen AddFlow
	mediaconnect: AddFlow VpcInterfaces
	mediaconect: CreateBridge
	mediaconect: CreateFlow
	mediaconect: CreateGateway
	mediaconect: DeleteBridge
	mediaconect: DeleteFlow
	mediaconect: DeleteGateway
	mediaconnect: Instanz DeregisterGateway
	mediaconect: DescribeBridge
	mediaconect: DescribeFlow
	mediaconect: DescribeFlow SourceMetadata
	mediaconect: DescribeGateway
	mediaconnect: Instanz DescribeGateway
	mediaconnect: DescribeOffering
	mediaconnect: DescribeReservation
	mediaconnect: Rechte GrantFlow

Servicepräfix	Aktionen
	mediaconnect: ListBridges
	mediaconect: ListEntitlements
	mediaconect: ListFlows
	mediaconect: Instanzen ListGateway
	mediaconect: ListGateways
	mediaconect: ListOfferings
	mediaconect: ListReservations
	mediaconect: PurchaseOffering
	mediaconect: Ausgabe RemoveBridge
	mediaconect: Quelle RemoveBridge
	mediaconect: RemoveFlow MediaStream
	mediaconect: Ausgabe RemoveFlow
	mediaconect: Quelle RemoveFlow
	mediaconect: RemoveFlow VpcInterface
	mediaconect: Anspruch RevokeFlow
	mediaconect: StartFlow
	mediaconect: StopFlow
	mediaconect: UpdateBridge
	mediaconect: Ausgabe UpdateBridge
	mediaconect: Quelle UpdateBridge
	mediaconect: Bundesstaat UpdateBridge

Servicepräfix	Aktionen
	mediaconnect: UpdateFlow
	mediaconnect: Anspruch UpdateFlow
	mediaconnect: UpdateFlow MediaStream
	mediaconnect: Ausgabe UpdateFlow
	mediaconnect: Quelle UpdateFlow
	mediaconnect: Instanz UpdateGateway

Servicepräfix	Aktionen
mediaconvert	mediaconvert: AssociateCertificate mediaconvert: CancelJob mediaconvert: CreateJob mediaconvert: Vorlage CreateJob mediaconvert: CreatePreset mediaconvert: CreateQueue mediaconvert: Vorlage DeleteJob mediaconvert: DeletePolicy mediaconvert: DeletePreset mediaconvert: DeleteQueue mediaconvert: DescribeEndpoints mediaconvert: DisassociateCertificate mediaconvert: GetJob mediaconvert: Vorlage GetJob mediaconvert: GetPolicy mediaconvert: GetPreset mediaconvert: GetQueue mediaconvert: ListJobs mediaconvert: Vorlagen ListJob mediaconvert: ListPresets mediaconvert: ListQueues

Servicepräfix	Aktionen
	mediaconvert: PutPolicy mediaconvert: Vorlage UpdateJob mediaconvert: UpdatePreset mediaconvert: UpdateQueue

Servicepräfix	Aktionen
medialive	medialiv: AcceptInput DeviceTransfer
	medial live: BatchDelete
	medial live: BatchStart
	medial live: BatchStop
	medialive: Zeitplan BatchUpdate
	medialive: CancelInput DeviceTransfer
	medial live: ClaimDevice
	medial live: CreateChannel
	medialive: Vorlage CreateCloud WatchAlarm
	medialive: CreateCloud WatchAlarm TemplateGroup
	medialive: Vorlage CreateEvent BridgeRule
	medialive: CreateEvent BridgeRule TemplateGroup
	medial live: CreateInput
	medial live: CreateInput SecurityGroup
	medial live: CreateMultiplex
	medialive: Programm CreateMultiplex
	medialive: Eingabe CreatePartner
	medialive: Karte CreateSignal
	medialive: DeleteChannel
	medialive: Vorlage DeleteCloud WatchAlarm
	medialive: DeleteCloud WatchAlarm TemplateGroup

Servicepräfix	Aktionen
	medialive: Vorlage DeleteEvent BridgeRule
	medialive: DeleteEvent BridgeRule TemplateGroup
	medial live: DeleteInput
	medial live: DeleteInput SecurityGroup
	medial live: DeleteMultiplex
	medialive: Programm DeleteMultiplex
	medialive: DeleteReservation
	medial live: DeleteSchedule
	medialive: Karte DeleteSignal
	medialive: Konfiguration DescribeAccount
	medialive: DescribeChannel
	medial live: DescribeInput
	medialive: Gerät DescribeInput
	medialive: DescribeInput DeviceThumbnail
	medial live: DescribeInput SecurityGroup
	medial live: DescribeMultiplex
	medialive: Programm DescribeMultiplex
	medialive: DescribeOffering
	medial live: DescribeReservation
	medial live: DescribeSchedule
	medial live: DescribeThumbnails

Servicepräfix	Aktionen
	medialive: Vorlage GetCloud WatchAlarm
	medialive: GetCloud WatchAlarm TemplateGroup
	medialive: Vorlage GetEvent BridgeRule
	medialive: GetEvent BridgeRule TemplateGroup
	medialive: Karte GetSignal
	medialive: ListChannels
	medial live: ListCloud WatchAlarm TemplateGroups
	medialive: Vorlagen ListCloud WatchAlarm
	medialive: ListEvent BridgeRule TemplateGroups
	medialive: Vorlagen ListEvent BridgeRule
	medialive: Geräte ListInput
	medialive: ListInput DeviceTransfers
	medial live: ListInputs
	medial live: ListInput SecurityGroups
	medial live: ListMultiplexes
	medialive: Programme ListMultiplex
	medialive: ListOfferings
	medial live: ListReservations
	medialive: Karten ListSignal
	medialive: PurchaseOffering
	medialive: Gerät RebootInput

Servicepräfix	Aktionen
	medialive: RejectInput DeviceTransfer
	medialive: Pipelines RestartChannel
	medialive: StartChannel
	medial live: StartDelete MonitorDeployment
	medialive: Gerät StartInput
	medialive: Fenster StartInput DeviceMaintenance
	medialive: Bereitstellung StartMonitor
	medialive: StartMultiplex
	medial live: StartUpdate SignalMap
	medial live: StopChannel
	medialive: Gerät StopInput
	medialive: StopMultiplex
	medialive: Gerät TransferInput
	medialive: Konfiguration UpdateAccount
	medialive: UpdateChannel
	medialive: Klasse UpdateChannel
	medialive: Vorlage UpdateCloud WatchAlarm
	medialive: UpdateCloud WatchAlarm TemplateGroup
	medialive: Vorlage UpdateEvent BridgeRule
	medialive: UpdateEvent BridgeRule TemplateGroup
	medial live: UpdateInput

Servicepräfix	Aktionen
	medialive: Gerät UpdateInput medialive: UpdateInput SecurityGroup medial live: UpdateMultiplex medialive: Programm UpdateMultiplex medialive: UpdateReservation

Servicepräfix	Aktionen
mediapackage	Medienpaket: ConfigureLogs Medienpaket: CreateChannel Medienpaket: Job CreateHarvest mediapackage: Endpunkt CreateOrigin Medienpaket: DeleteChannel Medienpaket: Endpunkt DeleteOrigin Medienpaket: DescribeChannel Medienpaket: Job DescribeHarvest mediapackage: Endpunkt DescribeOrigin Medienpaket: ListChannels Medienpaket: Jobs ListHarvest Medienpaket: Endpunkte ListOrigin Medienpaket: ListTags ForResource mediapackage: Anmeldeinformationen RotateChannel Medienpaket: RotateIngest EndpointCredentials Medienpaket: TagResource Medienpaket: UntagResource Medienpaket: UpdateChannel Medienpaket: Endpunkt UpdateOrigin

Servicepräfix	Aktionen
mediapackage-vod	Medienpaket-vod: ConfigureLogs Medienpaket-vod: CreateAsset mediapackage-vod: Konfiguration CreatePackaging mediapackage-vod: Gruppe CreatePackaging mediapaket-vod: DeleteAsset mediapackage-vod: Konfiguration DeletePackaging mediapackage-vod: Gruppe DeletePackaging mediapaket-vod: DescribeAsset mediapackage-vod: Konfiguration DescribePackaging mediapackage-vod: Gruppe DescribePackaging mediapaket-vod: ListAssets mediapackage-vod: Konfigurationen ListPackaging mediapackage-vod: Gruppen ListPackaging mediapaket-vod: ListTags ForResource Medienpaket-vod: TagResource Medienpaket-vod: UntagResource mediapackage-vod: Gruppe UpdatePackaging

Servicepräfix	Aktionen
mediastore	Mediastore: CreateContainer Medienspeicher: DeleteContainer mediastore: Richtlinie DeleteContainer mediastore: Richtlinie DeleteCors mediastore: Richtlinie DeleteLifecycle mediastore: Richtlinie DeleteMetric Mediastore: DescribeContainer mediastore: Richtlinie GetContainer mediastore: Richtlinie GetCors mediastore: Richtlinie GetLifecycle mediastore: Richtlinie GetMetric Mediastore: ListContainers mediastore: Richtlinie PutContainer mediastore: Richtlinie PutCors mediastore: Richtlinie PutLifecycle mediastore: Richtlinie PutMetric mediastore: Protokollierung StartAccess mediastore: Protokollierung StopAccess

Servicepräfix	Aktionen
mediatailor	mediatailor: Konfiguration ConfigureLogs ForPlayback
	mediatailor: CreateChannel
	mediatailor: Quelle CreateLive
	mediatailor: Zeitplan CreatePrefetch
	mediatailor: CreateProgram
	mediatailor: Standort CreateSource
	mediatailor: Quelle CreateVod
	mediatailor: DeleteChannel
	mediatailor: Richtlinie DeleteChannel
	mediatailor: Quelle DeleteLive
	mediatailor: Konfiguration DeletePlayback
	mediatailor: Zeitplan DeletePrefetch
	mediatailor: DeleteProgram
	mediatailor: Standort DeleteSource
	mediatailor: Quelle DeleteVod
	mediatailor: DescribeChannel
	mediatailor: Quelle DescribeLive
	mediatailor: DescribeProgram
	mediatailor: Standort DescribeSource
	mediatailor: Quelle DescribeVod
	mediatailor: Richtlinie GetChannel

Servicepräfix	Aktionen
	mediatailor: Zeitplan GetChannel
	mediatailor: Konfiguration GetPlayback
	mediatailor: Zeitplan GetPrefetch
	mediatailor: ListAlerts
	Mediatailor: ListChannels
	mediatailor: Quellen ListLive
	mediatailor: Konfigurationen ListPlayback
	mediatailor: Zeitpläne ListPrefetch
	mediatailor: Standorte ListSource
	mediatailor: Quellen ListVod
	mediatailor: Richtlinie PutChannel
	mediatailor: Konfiguration PutPlayback
	mediatailor: StartChannel
	Mediatailor: StopChannel
	Mediatailor: UpdateChannel
	mediatailor: Quelle UpdateLive
	mediatailor: UpdateProgram
	mediatailor: Standort UpdateSource
	mediatailor: Quelle UpdateVod

Servicepräfix	Aktionen
memorydb	memorydb: Cluster BatchUpdate Speicherdatenbank: CopySnapshot Speicher-DB: CreateAcl Speicher-DB: CreateCluster memorydb: Gruppe CreateParameter memorydb: CreateSnapshot memorydb: Gruppe CreateSubnet memorydb: CreateUser Speicher-DB: DeleteAcl Speicher-DB: DeleteCluster memorydb: Gruppe DeleteParameter memorydb: DeleteSnapshot memorydb: Gruppe DeleteSubnet memorydb: DeleteUser Speicher-DB: DescribeAcls Speicher-DB: DescribeClusters memorydb: Versionen DescribeEngine Speicherdatenbank: DescribeEvents memorydb: Gruppen DescribeParameter memorydb: DescribeParameters memorydb: Knoten DescribeReserved

Servicepräfix	Aktionen
	memorydb: DescribeReserved NodesOfferings memorydb: Aktualisierungen DescribeService Speicherdatenbank: DescribeSnapshots memorydb: Gruppen DescribeSubnet memorydb: DescribeUsers Speicher-DB: FailoverShard memorydb: Aktualisierungen ListAllowed NodeType Speicherdatenbank: PurchaseReserved NodesOffering memorydb: Gruppe ResetParameter memorydb: UpdateAcl Speicher-DB: UpdateCluster memorydb: Gruppe UpdateParameter memorydb: Gruppe UpdateSubnet memorydb: UpdateUser

Servicepräfix	Aktionen
mgh	mgh: Artifact AssociateCreated
	mgh: Ressource AssociateDiscovered
	mgh: CreateHome RegionControl
	mgh: CreateProgress UpdateStream
	mgh: DeleteHome RegionControl
	mgh: DeleteProgress UpdateStream
	mgh: Bundesland DescribeApplication
	mgh: DescribeHome RegionControls
	mgh: Aufgabe DescribeMigration
	mgh: Artifact DisassociateCreated
	mgh: Ressource DisassociateDiscovered
	mgh: Region GetHome
	mgh: Aufgabe ImportMigration
	mgh: Staaten ListApplication
	mgh: Artefakte ListCreated
	mgh: Ressourcen ListDiscovered
	mgh: Aufgaben ListMigration
	mgh: ListProgress UpdateStreams
	mgh: Bundesland NotifyApplication
	mgh: NotifyMigration TaskState
	mgh: Attribute PutResource

Servicepräfix	Aktionen
mgn	mgn: ArchiveApplication mg: ArchiveWave mg: AssociateApplications mgn: Server AssociateSource mgn: Bundesstaat ChangeServer LifeCycle mgn: CreateApplication mg: CreateConnector mg: CreateLaunch ConfigurationTemplate mg: CreateReplication ConfigurationTemplate mg: CreateWave mg: DeleteApplication mg: DeleteConnector mg: DeleteJob mg: DeleteLaunch ConfigurationTemplate mg: DeleteReplication ConfigurationTemplate mgn: Server DeleteSource mgn: Kunde DeleteVcenter mgn: DeleteWave mg: DescribeJob LogItems mg: DescribeJobs mg: DescribeLaunch ConfigurationTemplates

Servicepräfix	Aktionen
	mg: DescribeReplication ConfigurationTemplates mgn: Kunden DescribeVcenter mgn: DisassociateApplications mgn: Server DisassociateSource mgn: Dienst DisconnectFrom mgn: FinalizeCutover mgn: Konfiguration GetReplication mgn: InitializeService mg: ListConnectors mgn: Fehler ListExport mgn: ListExports mgn: Fehler ListImport mgn: ListImports mgn: Konten ListManaged mgn: ListSource ServerActions mgn: Aktionen ListTemplate mgn: Archiviert MarkAs mgn: PauseReplication mg: PutSource ServerAction mgn: Aktion PutTemplate mgn: RemoveSource ServerAction

Servicepräfix	Aktionen
	mgn: Aktion RemoveTemplate mgn: ResumeReplication mgn: Replikation RetryData mgn: StartCutover mg: StartExport mg: StartImport mg: StartReplication mg: StartTest mg: StopReplication mgn: Instanzen TerminateTarget mgn: UnarchiveApplication mg: UnarchiveWave mg: UpdateApplication mg: UpdateConnector mg: UpdateLaunch ConfigurationTemplate mgn: Konfiguration UpdateReplication mgn: UpdateReplication ConfigurationTemplate mgn: Server UpdateSource mgn: Typ UpdateSource ServerReplication mgn: UpdateWave

Servicepräfix	Aktionen
migrationhub-strategy	MigrationHub-Strategie: Muster GetAnti
	MigrationHub-Strategie: GetApplication ComponentDetails
	Strategie für den Migrationshub: GetApplication Component Strategies
	Strategie für den Migrationshub: GetAssessment
	Strategie für den Migrationshub: GetImport FileTask
	Strategie für den Migrationshub: GetLatest AssessmentId
	Strategie für den Migrationshub: GetMessage
	migrationhub-strategy: Einstellungen GetPortfolio
	migrationhub-strategy: GetPortfolio Zusammenfassung
	MigrationHub-Strategie: GetRecommendation ReportDetails
	MigrationHub-Strategie: Einzelheiten GetServer
	Migrationhub-Strategie: GetServer Strategien
	ListAnalyzableMigrationHub-Strategie: Server
	MigrationHub-Strategie: ListAnti Muster
	MigrationHub-Strategie: ListApplication Komponenten
	MigrationHub-Strategie: ListCollectors
	Strategie für den Migrationshub: ListImport FileTask
	MigrationHub-Strategie: ListJar Artefakte
	MigrationHub-Strategie: ListServers
	migrationhub-strategy: Einstellungen PutPortfolio

Servicepräfix	Aktionen
	MigrationHub-Strategie: RegisterCollector
	MigrationHub-Strategie: SendMessage
	MigrationHub-Strategie: StartAssessment
	Strategie für den Migrationshub: StartImport FileTask
	Strategie für den Migrationshub: StartRecommendation ReportGeneration
	Strategie für den Migrationshub: StopAssessment
	Strategie für den Migrationshub: UpdateApplication ComponentConfig
	migrationhub-strategy: Konfiguration UpdateCollector
	migrationhub-strategy: Config UpdateServer

Servicepräfix	Aktionen
mobiletargeting	Mobiles Targeting: CreateApp Zielgruppenansprache auf Mobilgeräten: CreateCampaign mobiletargeting: Vorlage CreateEmail Mobiletargeting: Job CreateExport Mobiletargeting: Job CreateImport mobiles Targeting: CreateIn AppTemplate Zielgruppenansprache auf Mobilgeräten: CreateJourney mobiletargeting: Vorlage CreatePush mobiletargeting: CreateRecommender Konfiguration Mobiletargeting: CreateSegment mobiletargeting: Vorlage CreateSms mobiletargeting: CreateVoice Vorlage mobiletargeting: DeleteAdm Kanal mobiletargeting: DeleteApns Kanal mobiles Targeting: DeleteApns SandboxChannel Zielgruppenansprache auf Mobilgeräten: DeleteApns VoipChannel mobiletargeting: Kanal DeleteApns VoipSandbox mobiles Targeting: DeleteApp mobiletargeting: Kanal DeleteBaidu mobiles Targeting: DeleteCampaign mobiletargeting: Kanal DeleteEmail

Servicepräfix	Aktionen
	mobiletargeting: DeleteEmail Vorlage Mobiletargeting: DeleteEndpoint mobiles Targeting: Stream DeleteEvent mobiletargeting: DeleteGcm Kanal mobiles Targeting: DeleteIn AppTemplate Zielgruppenansprache auf Mobilgeräten: DeleteJourney mobiletargeting: Vorlage DeletePush mobiletargeting: DeleteRecommender Konfiguration Mobiletargeting: DeleteSegment mobiletargeting: Kanal DeleteSms mobiletargeting: DeleteSms Vorlage mobiletargeting: DeleteUser Endpunkte mobiletargeting: DeleteVoice Kanal mobiletargeting: DeleteVoice Vorlage mobiletargeting: GetAdm Kanal mobiletargeting: GetApns Kanal mobiles Targeting: GetApns SandboxChannel Zielgruppenansprache auf Mobilgeräten: GetApns VoipChannel mobiletargeting: Kanal GetApns VoipSandbox mobiles Targeting: GetApp

Servicepräfix	Aktionen
	Zielgruppenansprache auf Mobilgeräten: GetApplication DateRange Kpi mobiletargeting: GetApplication Einstellungen Mobiletargeting: GetApps mobiletargeting: Kanal GetBaidu mobiles Targeting: GetCampaign mobiletargeting: Aktivitäten GetCampaign Mobiletargeting: GetCampaign DateRange Kpi Zielgruppenansprache auf Mobilgeräten: GetCampaigns mobiles Targeting: Version GetCampaign mobiletargeting: GetCampaign Versionen Mobiletargeting: GetChannels mobiletargeting: Kanal GetEmail mobiletargeting: GetEmail Vorlage Mobiletargeting: GetEndpoint mobiles Targeting: Stream GetEvent Mobiletargeting: Job GetExport mobiletargeting: GetExport Jobs mobiletargeting: GetGcm Kanal Mobiletargeting: Job GetImport mobiletargeting: GetImport Jobs

Servicepräfix	Aktionen
	mobiles Targeting: GetIn AppMessages Zielgruppenansprache auf Mobilgeräten: GetIn AppTemplate Zielgruppenansprache auf Mobilgeräten: GetJourney Zielgruppenansprache auf Mobilgeräten: GetJourney DateRange Kpi Mobiletargeting: GetJourney ExecutionActivity Metriken Zielgruppenansprache auf Mobilgeräten: GetJourney Execution Metrics Zielgruppenansprache auf Mobilgeräten: GetJourney RunExecution ActivityMetrics Mobiletargeting: Metriken GetJourney RunExecution mobiletargeting: GetJourney Läuft mobiletargeting: GetPush Vorlage mobiletargeting: GetRecommender Konfiguration mobiletargeting: GetRecommender Konfigurationen mobiles Targeting: GetSegment Zielgruppenansprache auf Mobilgeräten: GetSegment ExportJobs Zielgruppenansprache auf Mobilgeräten: GetSegment ImportJobs Zielgruppenansprache auf Mobilgeräten: GetSegments mobiles Targeting: Version GetSegment mobiletargeting: GetSegment Versionen mobiletargeting: GetSms Kanal

Servicepräfix	Aktionen
	mobiletargeting: GetSms Vorlage mobiletargeting: GetUser Endpunkte mobiletargeting: GetVoice Kanal mobiletargeting: GetVoice Vorlage Mobiletargeting: ListJourneys Zielgruppenansprache auf Mobilgeräten: ListTemplates mobiletargeting: Versionen ListTemplate mobiletargeting: PhoneNumber Bestätigen mobiletargeting: PutEvent Stream mobiles Targeting: RemoveAttributes mobiletargeting: Kanal UpdateAdm mobiletargeting: UpdateApns Kanal mobiles Targeting: UpdateApns SandboxChannel Zielgruppenansprache auf Mobilgeräten: UpdateApns VoipChannel mobiletargeting: Kanal UpdateApns VoipSandbox mobiletargeting: UpdateApplication Einstellungen mobiletargeting: UpdateBaidu Kanal mobiles Targeting: UpdateCampaign mobiletargeting: Kanal UpdateEmail mobiletargeting: UpdateEmail Vorlage Mobiletargeting: UpdateEndpoint

Servicepräfix	Aktionen
	Mobiles Targeting: Batch UpdateEndpoints mobiletargeting: UpdateGcm Kanal mobiles Targeting: UpdateIn AppTemplate Zielgruppenansprache auf Mobilgeräten: UpdateJourney Mobiletargeting: Bundesland UpdateJourney mobiletargeting: UpdatePush Vorlage mobiletargeting: UpdateRecommender Konfiguration Mobiletargeting: UpdateSegment mobiletargeting: Kanal UpdateSms mobiletargeting: UpdateSms Vorlage Mobiletargeting: UpdateTemplate ActiveVersion mobiletargeting: Kanal UpdateVoice mobiletargeting: UpdateVoice Vorlage mobiletargeting:VerifyOTPMessage

Servicepräfix	Aktionen
mq	mq: CreateBroker mq: CreateConfiguration mq: CreateUser mq: DeleteBroker mq: DeleteUser mq: DescribeBroker mq: DescribeBroker EngineTypes mq: DescribeBroker InstanceOptions mq: DescribeConfiguration mq: Überarbeitung DescribeConfiguration mq: DescribeUser mq: ListBrokers mq: Überarbeitungen ListConfiguration mq: ListConfigurations mq: ListUsers mq:Promote mq: RebootBroker mq: UpdateBroker mq: UpdateConfiguration mq: UpdateUser

Servicepräfix	Aktionen
networkmanager	Netzwerkmanager: AcceptAttachment Netzwerkmanager: Peer AssociateConnect Netzwerkmanager: Gateway AssociateCustomer Netzwerkmanager: AssociateLink Netzwerkmanager: Peer AssociateTransit GatewayConnect networkmanager: Anlage CreateConnect Netzwerkmanager: CreateConnection Netzwerkmanager: Peer CreateConnect networkmanager: Netzwerk CreateCore Netzwerkmanager: CreateDevice networkmanager: Netzwerk CreateGlobal Netzwerkmanager: CreateLink Netzwerkmanager: CreateSite Netzwerkmanager: CreateSite ToSite VpnAttachment Netzwerkmanager: CreateTransit GatewayPeering Netzwerkmanager: CreateTransit GatewayRoute TableAttachment networkmanager: Anlage CreateVpc Netzwerkmanager: DeleteAttachment Netzwerkmanager: DeleteConnection Netzwerkmanager: Peer DeleteConnect networkmanager: Netzwerk DeleteCore

Servicepräfix	Aktionen
	Netzwerkmanager: Version DeleteCore NetworkPolicy Netzwerkmanager: DeleteDevice networkmanager: Netzwerk DeleteGlobal Netzwerkmanager: DeleteLink Netzwerkmanager: DeletePeering networkmanager: Richtlinie DeleteResource Netzwerkmanager: DeleteSite Netzwerkmanager: Gateway DeregisterTransit networkmanager: Netzwerke DescribeGlobal Netzwerkmanager: Peer DisassociateConnect Netzwerkmanager: Gateway DisassociateCustomer Netzwerkmanager: DisassociateLink Netzwerkmanager: Peer DisassociateTransit GatewayConnect Netzwerkmanager: Eingestellt ExecuteCore NetworkChange networkmanager: Anhang GetConnect Netzwerkmanager: GetConnections Netzwerkmanager: Peer GetConnect Netzwerkmanager: GetConnect PeerAssociations networkmanager: Netzwerk GetCore networkmanager: Ereignisse GetCore NetworkChange networkmanager: Satz GetCore NetworkChange

Servicepräfix	Aktionen
	Netzwerkmanager: GetCore NetworkPolicy Netzwerkmanager: GetCustomer GatewayAssociations Netzwerkmanager: GetDevices networkmanager: Verbände GetLink Netzwerkmanager: GetLinks Netzwerkmanager: GetNetwork ResourceCounts Netzwerkmanager: GetNetwork ResourceRelationships networkmanager: Ressourcen GetNetwork networkmanager: Routen GetNetwork networkmanager: Telemetrie GetNetwork networkmanager: Richtlinie GetResource networkmanager: Analyse GetRoute Netzwerkmanager: GetSites Netzwerkmanager: GetSite ToSite VpnAttachment Netzwerkmanager: GetTransit GatewayConnect PeerAssociations Netzwerkmanager: GetTransit GatewayPeering Netzwerkmanager: GetTransit GatewayRegistrations Netzwerkmanager: GetTransit GatewayRoute TableAttachment networkmanager: Anlage GetVpc Netzwerkmanager: ListAttachments Netzwerkmanager: Kollegen ListConnect

Servicepräfix	Aktionen
	networkmanager: Versionen ListCore NetworkPolicy networkmanager: Netzwerke ListCore Netzwerkmanager: Status ListOrganization ServiceAccess Netzwerkmanager: ListPeerings Netzwerkmanager: PutCore NetworkPolicy networkmanager: Richtlinie PutResource Netzwerkmanager: Gateway RegisterTransit Netzwerkmanager: RejectAttachment Netzwerkmanager: Version RestoreCore NetworkPolicy networkmanager: Aktualisieren StartOrganization ServiceAccess networkmanager: Analyse StartRoute Netzwerkmanager: UpdateConnection networkmanager: Netzwerk UpdateCore Netzwerkmanager: UpdateDevice networkmanager: Netzwerk UpdateGlobal Netzwerkmanager: UpdateLink Netzwerkmanager: UpdateNetwork ResourceMetadata Netzwerkmanager: UpdateSite networkmanager: Anlage UpdateVpc

Servicepräfix	Aktionen
nimble	flink: AcceptEulas nimble: Profil CreateLaunch nimble: Bild CreateStreaming nimble: Sitzung CreateStreaming flink: CreateStreaming SessionStream flink: CreateStudio nimble: Komponente CreateStudio nimble: Profil DeleteLaunch flink: DeleteLaunch ProfileMember nimble: Bild DeleteStreaming nimble: Sitzung DeleteStreaming flink: DeleteStudio nimble: Komponente DeleteStudio nimble: Mitglied DeleteStudio nimble: GetEula flink: GetLaunch ProfileDetails nimble: Bild GetStreaming nimble: Sitzung GetStreaming flink: GetStreaming SessionBackup flink: GetStreaming SessionStream flink: GetStudio

Servicepräfix	Aktionen
	nimble: Komponente GetStudio
	nimble: Mitglied GetStudio
	nimble: ListEulas
	flink: ListLaunch ProfileMembers
	nimble: Profile ListLaunch
	nimble: Bilder ListStreaming
	flink: ListStreaming SessionBackups
	nimble: Sitzungen ListStreaming
	nimble: Komponenten ListStudio
	nimble: Mitglieder ListStudio
	flink: ListStudios
	flink: PutLaunch ProfileMembers
	nimble: Mitglieder PutStudio
	nimble: Sitzung StartStreaming
	nimble: SSO StartStudio ConfigurationRepair
	nimble: Sitzung StopStreaming
	nimble: Profil UpdateLaunch
	flink: UpdateLaunch ProfileMember
	nimble: Bild UpdateStreaming
	flink: UpdateStudio
	nimble: Komponente UpdateStudio

Servicepräfix	Aktionen
omics	Comics: Hochladen AbortMultipart ReadSet Comics: BatchDelete ReadSet Comics: CancelAnnotation ImportJob Comics: CancelRun Comics: CancelVariant ImportJob Omics: Hochladen CompleteMultipart ReadSet Omics: Speichern CreateAnnotation Omics: Hochladen CreateMultipart ReadSet Omics: Speichern CreateReference Omics: Gruppe CreateRun Omics: Geschäft CreateSequence Omics: Geschäft CreateVariant Comics: CreateWorkflow Comics: Geschäft DeleteAnnotation Comics: DeleteReference Comics: Geschäft DeleteReference Comics: DeleteRun Omics: Gruppe DeleteRun Omics: Geschäft DeleteSequence Omics: Geschäft DeleteVariant Comics: DeleteWorkflow

Servicepräfix	Aktionen
	Comics: GetAnnotation ImportJob Comics: Geschäft GetAnnotation Comics: Set GetRead Comics: Job GetRead SetActivation Comics: Job GetRead SetExport Comics: Job GetRead SetImport Comics: GetRead SetMetadata Comics: GetReference Comics: GetReference ImportJob Omics: Metadaten GetReference Omics: Speichern GetReference Comics: GetRun Omics: Gruppe GetRun Omics: Aufgabe GetRun Omics: Geschäft GetSequence Comics: GetVariant ImportJob Comics: Geschäft GetVariant Comics: GetWorkflow Comics: ListAnnotation ImportJobs Comics: Geschichten ListAnnotation Omics: Uploads ListMultipart ReadSet

Servicepräfix	Aktionen
	Omics: Jobs ListRead SetActivation Omics: Jobs ListRead SetExport Omics: Jobs ListRead SetImport Comics: Sets ListRead Comics: Teile ListRead SetUpload Comics: ListReference ImportJobs Comics: ListReferences Comics: Geschichten ListReference Omics: Gruppen ListRun Omics: ListRuns Omics: Aufgaben ListRun Omics: Geschichten ListSequence Comics: ListVariant ImportJobs Comics: Geschichten ListVariant Comics: ListWorkflows Comics: StartAnnotation ImportJob Comics: Job StartRead SetActivation Comics: Job StartRead SetExport Comics: Job StartRead SetImport Comics: StartReference ImportJob Comics: StartRun

Servicepräfix	Aktionen
	Comics: StartVariant ImportJob Comics: Geschäft UpdateAnnotation Omics: Gruppe UpdateRun Omics: Geschäft UpdateVariant Comics: UpdateWorkflow Comics: UploadRead SetPart

Servicepräfix	Aktionen
opsworks	Opsworks: AssignInstance
	Opsworks: AssignVolume
	opsworks: IP AssociateElastic
	opsworks: AttachElastic LoadBalancer
	Opsworks: CloneStack
	Opsworks: CreateApp
	Opsworks: CreateDeployment
	Opsworks: CreateInstance
	Opsworks: CreateLayer
	Opsworks: CreateStack
	opsworks: Profil CreateUser
	opsworks: DeleteApp
	Opsworks: DeleteInstance
	Opsworks: DeleteLayer
	Opsworks: DeleteStack
	opsworks: Profil DeleteUser
	opsworks: Cluster DeregisterEcs
	opsworks: IP DeregisterElastic
	opsworks: DeregisterInstance
	Opsworks: DeregisterRds DbInstance
	Opsworks: DeregisterVolume

Servicepräfix	Aktionen
	opsworks: Versionen DescribeAgent
	opsworks: DescribeApps
	Opsworks: DescribeCommands
	Opsworks: DescribeDeployments
	opsworks: Cluster DescribeEcs
	opsworks: Ips DescribeElastic
	opsworks: DescribeElastic LoadBalancers
	Opsworks: DescribeInstances
	Opsworks: DescribeLayers
	opsworks: Skalierung DescribeLoad BasedAuto
	opsworks: DescribeMy UserProfile
	opsworks: Systeme DescribeOperating
	opsworks: DescribePermissions
	opsworks: Arrays DescribeRaid
	opsworks: DescribeRds DbInstances
	opsworks: Fehler DescribeService
	opsworks: DescribeStack ProvisioningParameters
	Opsworks: DescribeStacks
	opsworks: Zusammenfassung DescribeStack
	opsworks: Skalierung DescribeTime BasedAuto
	opsworks: Profile DescribeUser

Servicepräfix	Aktionen
	opsworks: DescribeVolumes
	Opsworks: DetachElastic LoadBalancer
	opsworks: IP DisassociateElastic
	opsworks: Vorschlag GetHostname
	opsworks: GrantAccess
	Opsworks: RebootInstance
	opsworks: Cluster RegisterEcs
	opsworks: IP RegisterElastic
	opsworks: RegisterInstance
	Opsworks: RegisterRds DbInstance
	Opsworks: RegisterVolume
	opsworks: Skalierung SetLoad BasedAuto
	opsworks: SetPermission
	opsworks: Skalierung SetTime BasedAuto
	opsworks: StartInstance
	Opsworks: StartStack
	Opsworks: StopInstance
	Opsworks: StopStack
	Opsworks: UnassignInstance
	Opsworks: UnassignVolume
	Opsworks: UpdateApp

Servicepräfix	Aktionen
	opsworks: IP UpdateElastic
	opsworks: UpdateInstance
	Opsworks: UpdateLayer
	Opsworks: UpdateMy UserProfile
	Opsworks: UpdateRds DbInstance
	Opsworks: UpdateStack
	opsworks: Profil UpdateUser
	opsworks: UpdateVolume

Servicepräfix	Aktionen
opsworks-cm	opsworks-cm: AssociateNode opsworks-cm: CreateBackup opsworks-cm: CreateServer opsworks-cm: DeleteBackup opsworks-cm: DeleteServer opsworks-cm: Attribute DescribeAccount opsworks-cm: DescribeBackups opsworks-cm: DescribeEvents opsworks-cm: DescribeNode AssociationStatus opsworks-cm: DescribeServers opsworks-cm: DisassociateNode opsworks-cm: ExportServer EngineAttribute opsworks-cm: RestoreServer opsworks-cm: StartMaintenance opsworks-cm: UpdateServer opsworks-cm: UpdateServer EngineAttributes

Servicepräfix	Aktionen
Organisationen	Organisationen: AcceptHandshake
	Organisationen: AttachPolicy
	Organisationen: CancelHandshake
	Organisationen: CloseAccount
	Organisationen: CreateAccount
	Organisationen: CreateGov CloudAccount
	Organisationen: CreateOrganization
	Organisationen: CreateOrganizational Einheit
	Organisationen: CreatePolicy
	Organisationen: DeclineHandshake
	Organisationen: DeleteOrganization
	Organisationen: DeleteOrganizational Einheit
	Organisationen: DeletePolicy
	Organisationen: DeleteResource Politik
	Organisationen: DeregisterDelegated Administrator
	Organisationen: DescribeAccount
	Organisationen: DescribeCreate AccountStatus
	Organisationen: DescribeEffective Politik
	Organisationen: DescribeHandshake
	Organisationen: DescribeOrganization
	Organisationen: DescribeOrganizational Einheit

Servicepräfix	Aktionen
	Organisationen: DescribePolicy
	Organisationen: DescribeResource Politik
	Organisationen: DetachPolicy
	Organisationen: deaktivieren AWSServiceAccess
	Organisationen: Typ DisablePolicy
	Organisationen: EnableAll Funktionen
	Organisationen: aktivieren AWSServiceAccess
	Organisationen: Typ EnablePolicy
	Organisationen: InviteAccount ToOrganization
	Organisationen: LeaveOrganization
	Organisationen: ListAccounts
	Organisationen: ListAccounts ForParent
	Organisationen: Liste AWSServiceAccessForOrganization
	Organisationen: ListChildren
	Organisationen: ListCreate AccountStatus
	Organisationen: ListDelegated Administratoren
	Organisationen: ListDelegated ServicesFor Konto
	Organisationen: ListHandshakes ForAccount
	Organisationen: ListHandshakes ForOrganization
	Organisationen: ListOrganizational UnitsFor Muttergesellschaft
	Organisationen: ListParents

Servicepräfix	Aktionen
	Organisationen: ListPolicies
	Organisationen: ListPolicies ForTarget
	Organisationen: ListRoots
	Organisationen: ListTargets ForPolicy
	Organisationen: MoveAccount
	Organisationen: PutResource Politik
	Organisationen: RegisterDelegated Administrator
	Organisationen: RemoveAccount FromOrganization
	Organisationen: UpdateOrganizational Einheit
	Organisationen: UpdatePolicy

Servicepräfix	Aktionen
outposts	Außenposten: Aufgabe CancelCapacity Außenposten: CancelOrder Außenposten: CreateOrder Außenposten: CreateOutpost Außenposten: CreatePrivate ConnectivityConfig Außenposten: CreateSite Außenposten: DeleteOutpost Außenposten: DeleteSite Außenposten: Aufgabe GetCapacity Außenposten: Gegenstand GetCatalog Außenposten: GetConnection Außenposten: GetOrder Außenposten: GetOutpost Außenposten: GetOutpost InstanceTypes Außenposten: Typen GetOutpost SupportedInstance Außenposten: GetPrivate ConnectivityConfig Außenposten: GetSite Außenposten: Adresse GetSite Außenposten: ListAssets Außenposten: Aufgaben ListCapacity Außenposten: Gegenstände ListCatalog

Servicepräfix	Aktionen
	Außenposten: ListOrders Außenposten: ListOutposts Außenposten: ListSites Außenposten: Aufgabe StartCapacity Außenposten: StartConnection Außenposten: UpdateOutpost Außenposten: UpdateSite Außenposten: Adresse UpdateSite Außenposten: Immobilien UpdateSite RackPhysical

Servicepräfix	Aktionen
panorama	Panorama: Instanz CreateApplication
	Panorama: CreateJob ForDevices
	Panorama: CreateNode FromTemplate Job
	Panorama: CreatePackage
	Panoramabild: CreatePackage ImportJob
	Panoramabild: DeleteDevice
	Panorama: DeletePackage
	Panorama: DeregisterPackage Ausführung
	Panorama: DescribeApplication Instanz
	Panorama: DescribeApplication InstanceDetails
	Panoramabild: DescribeDevice
	Panorama: DescribeDevice Job
	Panorama: DescribeNode
	Panorama: DescribeNode FromTemplate Job
	Panorama: DescribePackage
	Panoramabild: DescribePackage ImportJob
	Panorama: DescribePackage Ausführung
	Panorama: ListApplication InstanceDependencies
	Panorama: ListApplication InstanceNode Instanzen
	Panorama: ListApplication Instanzen
	Panorama: ListDevices

Servicepräfix	Aktionen
	Panorama: ListDevices Jobs Panorama: ListNode FromTemplate Jobs Panorama: ListNodes Panoramabild: ListPackage ImportJobs Panoramabild: ListPackages Panorama: ProvisionDevice Panorama: RegisterPackage Ausführung Panorama: RemoveApplication Instanz Panorama: SignalApplication InstanceNode Instanzen Panorama: UpdateDevice Metadaten
pi	pi: CreatePerformance AnalysisReport Pi: DeletePerformance AnalysisReport pi: DescribeDimension Schlüssel pi: GetDimension KeyDetails Pi: GetPerformance AnalysisReport pi: GetResource Metadaten pi: GetResource Metriken pi: ListAvailable ResourceDimensions Pi: ListAvailable ResourceMetrics Pi: ListPerformance AnalysisReports

Servicepräfix	Aktionen
pipes	Rohre: CreatePipe Rohre: DeletePipe Rohre: DescribePipe Rohre: ListPipes Rohre: StartPipe Rohre: StopPipe Rohre: UpdatePipe
polly	Polly: DeleteLexicon Polly: DescribeVoices Polly: GetLexicon Polly: GetSpeech SynthesisTask Polly: ListLexicons Polly: ListSpeech SynthesisTasks Polly: PutLexicon Polly: StartSpeech SynthesisTask Polly: SynthesizeSpeech

Servicepräfix	Aktionen
Profil	Profil: Key AddProfile
	Profil: CreateCalculated AttributeDefinition
	profil: CreateDomain
	Profil: CreateEvent Stream
	Profil: CreateProfile
	profil: DeleteCalculated AttributeDefinition
	profil: DeleteDomain
	Profil: DeleteEvent Stream
	Profil: DeleteIntegration
	profil: DeleteProfile
	Profil: DeleteProfile Schlüssel
	Profil: DeleteProfile Objekt
	Profil: DeleteProfile ObjectType
	profil: DeleteWorkflow
	profil: DetectProfile ObjectType
	profil: GetAuto MergingPreview
	profil: GetCalculated AttributeDefinition
	Profil: GetCalculated AttributeFor Profil
	Profil: GetDomain
	Profil: GetEvent Stream
	Profil: GetIdentity ResolutionJob

Servicepräfix	Aktionen
	profil: GetIntegration
	profil: GetMatches
	profil: GetProfile ObjectType
	Profil: GetProfile ObjectType Vorlage
	Profil: GetSimilar Profile
	Profil: GetWorkflow
	Profil: GetWorkflow Steps
	Profil: ListAccount Integrationen
	Profil: ListCalculated AttributeDefinitions
	Profil: ListCalculated AttributesFor Profil
	Profil: ListDomains
	Profil: ListEvent Streams
	Profil: ListIdentity ResolutionJobs
	profil: ListIntegrations
	Profil: ListProfile Objekte
	Profil: ListProfile ObjectTypes
	Profil: ListProfile ObjectType Vorlagen
	Profil: ListRule BasedMatches
	profil: ListWorkflows
	profil: MergeProfiles
	profil: PutIntegration

Servicepräfix	Aktionen
	Profil: PutProfile Objekt Profil: PutProfile ObjectType profil: SearchProfiles profil: UpdateCalculated AttributeDefinition profil: UpdateDomain profil: UpdateProfile

Servicepräfix	Aktionen
qldb	qldb: CancelJournal KinesisStream qldb: CreateLedger qldb: DeleteLedger qldb: DescribeJournal KinesisStream qldb: S3 exportieren DescribeJournal qldb: DescribeLedger qldb: bis S3 ExportJournal qldb: GetBlock qldb: GetDigest qldb: GetRevision qldb: ListJournal KinesisStreams ForLedger qldb: S3-Exporte ListJournal qldb: S3-Hauptbuch ListJournal ExportsFor qldb: ListLedgers qldb: StreamJournal ToKinesis qldb: UpdateLedger qldb: UpdateLedger PermissionsMode

Servicepräfix	Aktionen
ram	ram: AcceptResource ShareInvitation
	ram: AssociateResource Teilen
	ram: AssociateResource SharePermission
	RAM: CreatePermission
	RAM: CreatePermission Ausführung
	ram: CreateResource Teilen
	ram: DeletePermission
	RAM: DeletePermission Ausführung
	ram: DeleteResource Teilen
	ram: DisassociateResource Teilen
	ram: DisassociateResource SharePermission
	ram: EnableSharing WithAws Organisation
	ram: GetPermission
	ram: GetResource Richtlinien
	ram: GetResource ShareAssociations
	RAM: GetResource ShareInvitations
	ram: GetResource Aktien
	RAM: ListPending InvitationResources
	ram: ListPermission Verbände
	ram: ListPermissions
	ram: ListPermission Versionen

Servicepräfix	Aktionen
	ram: ListPrincipals ram: ListReplace PermissionAssociations Arbeit RAM: ListResources RAM: ListResource SharePermissions ram: ListResource Typen ram: PromotePermission CreatedFrom Richtlinie ram: PromoteResource ShareCreated FromPolicy RAM: RejectResource ShareInvitation ram: ReplacePermission Verbände ram: SetDefault PermissionVersion ram: UpdateResource Teilen
rbin	Robin: CreateRule Ablage: DeleteRule Ablage: GetRule Ablage: ListRules Ablage: LockRule Ablage: UnlockRule Ablage: UpdateRule

Servicepräfix	Aktionen
rds	rds: AddRole zu DB-Cluster
	rds: ToDB-Instanz AddRole
	rds: Abonnement AddSource IdentifierTo
	rds: ApplyPending MaintenanceAction
	rdsSecurityGroup: Autorisierter DB-Zugriff
	rds:BacktrackDBCluster
	rds: Aufgabe CancelExport
	rdsClusterParameter: CopyDB-Gruppe
	rds: CopyDB ClusterSnapshot
	RDS: CopyDB ParameterGroup
	rds:CopyDBSnapshot
	rds: Gruppe CopyOption
	rds: CreateCustom DB EngineVersion
	rdsClusterParameter: DB-Gruppe erstellt
	rds:CreateDB ClusterSnapshot
	rds:CreatedB ParameterGroup
	rds:CreateDBProxy
	rds:CreatedB ProxyEndpoint
	rds:CreatedB SecurityGroup
	rds:CreateDBSnapshot
	rds:CreatedB SubnetGroup

Servicepräfix	Aktionen
	rds: Abonnement CreateEvent
	rds: CreateGlobal Cluster
	rds: CreateOption Gruppe
	rds: DeleteBlue GreenDeployment
	RDSClusterAutomated: gelöschtes DB-Backup
	rds: DeletedB-Gruppe ClusterParameter
	rds:DeletedB ClusterSnapshot
	RDSInstanceAutomated: gelöschtes DB-Backup
	RDS: Gelöschte Datenbank ParameterGroup
	rds:DeleteDBProxy
	rds: DeletedB ProxyEndpoint
	rds: DeletedB SecurityGroup
	rds:DeleteDBSnapshot
	rds: DeletedB SubnetGroup
	rds: Abonnement DeleteEvent
	rds: DeleteGlobal Cluster
	rds: DeleteOption Gruppe
	rds: DeregisterDB ProxyTargets
	rds: Attribute DescribeAccount
	rds: DescribeBlue GreenDeployments
	rds: DescribeCertificates

Servicepräfix	Aktionen
	RDSClusterAutomated: Beschriebene DB-Backups
	RDS: DescribeDB ClusterBacktracks
	rds: DescribeDB ClusterEndpoints
	rds:Beschriebene DB-Gruppen ClusterParameter
	RDS: DescribeDB ClusterParameters
	rds:DescribeDBClusters
	rds:DescribeDB-Attribute ClusterSnapshot
	rds:DescribeDB ClusterSnapshots
	rds: DescribeDB EngineVersions
	rds: InstanceAutomated Beschriebene DB-Backups
	rds:DescribeDBInstances
	RDS: DescribeDB LogFiles
	rds: DescribeDB ParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds: DescribeDB ProxyEndpoints
	rds:Beschriebene DB-Gruppen ProxyTarget
	RDS: DescribeDB ProxyTargets
	RDS: Beschriebene DB-Empfehlungen
	RDS: Beschrieben von B SecurityGroups
	rds: DescribeDB SnapshotAttributes

Servicepräfix	Aktionen
	rds:DescribeDBSnapshots
	rds: Datenbanken DescribeDb SnapshotTenant
	rds: Described B SubnetGroups
	rds: Parameter DescribeEngine DefaultCluster
	rds: DescribeEngine DefaultParameters
	rds: DescribeEvent Kategorien
	rds: DescribeEvents
	rds: DescribeEvent Abonnements
	rds: DescribeExport Aufgaben
	rds: DescribeGlobal Cluster
	rds: DescribeIntegrations
	rds: DescribeOption GroupOptions
	rds: DescribeOption Gruppen
	rds: DescribeOrderable DB InstanceOptions
	rds: DescribePending MaintenanceActions
	rds: DescribeReserved DB-Instances
	rds: DB DescribeReserved InstancesOfferings
	rds: DescribeSource Regionen
	rds: DescribeTenant Datenbanken
	rds: DescribeValid DB InstanceModifications
	rds: DownloadComplete DB LogFile

Servicepräfix	Aktionen
	rdsLogFile: DB-Teil herunterladen
	rds:FailoverDBCluster
	rds: Cluster FailoverGlobal
	rds: ModifyActivity Streamen
	rds: ModifyCertificates
	rds: ModifyCurrent DB ClusterCapacity
	rds: DB ändern ClusterEndpoint
	rds:DB-Gruppe modifizieren ClusterParameter
	ClusterSnapshotrds:ModifyDB-Attribut
	rds:ModifyDB ParameterGroup
	rds:ModifyDBProxy
	rds: DB modifizieren ProxyEndpoint
	rds:DB-Gruppe modifizieren ProxyTarget
	rds: DB-Empfehlung modifizieren
	rds:ModifyDBSnapshot
	RDS: DB modifizieren SnapshotAttribute
	rds: DB modifizieren SubnetGroup
	rds: Abonnement ModifyEvent
	rds: ModifyGlobal Cluster
	rds: ModifyOption Gruppe
	rds: ModifyTenant Datenbank

Servicepräfix	Aktionen
	rds: PurchaseReserved DB InstancesOffering
	rds:RebootDBCluster
	rds: RegisterDB ProxyTargets
	rds: RemoveFrom GlobalCluster
	rds: RemoveRole aus DBCluster
	rds: FromDBInstance RemoveRole
	rds: Abonnement RemoveSource IdentifierFrom
	rdsClusterParameter: ResetDB-Gruppe
	rds: ResetDB ParameterGroup
	rds: Datenbank S3 wiederhergestellt ClusterFrom
	rds: ClusterFrom Wiederhergestellter DB-Snapshot
	rds: Wiederhergestellte DB-Zeit ClusterTo PointIn
	rds: Der DB-DB-Snapshot wurde wiederhergestellt InstanceFrom
	rds: RestoreDB S3 InstanceFrom
	rds: Wiederhergestellte DB-Zeit InstanceTo PointIn
	rds: Widerrufener DB-Eingang SecurityGroup
	rds: Streamen StartActivity
	rds:StartDBCluster
	rds:StartDBInstance
	rds: StartDB InstanceAutomated BackupsReplication
	rds: Aufgabe StartExport

Servicepräfix	Aktionen
	rds: StopActivity Streamen rds:StopDBCluster rds:StopDBInstance RDS: StopDB InstanceAutomated BackupsReplication rds: SwitchoverBlue GreenDeployment rds: SwitchoverGlobal Cluster rds: SwitchoverRead Replikat

Servicepräfix	Aktionen
redshift	Rotverschiebung: AcceptReserved NodeExchange Rotverschiebung: AddPartner Rotverschiebung: AssociateData ShareConsumer Rotverschiebung: Eintritt AuthorizeCluster SecurityGroup redshift: Teilen AuthorizeData redshift: Zugriff AuthorizeEndpoint redshift: Zugriff AuthorizeSnapshot Redshift: BatchDelete ClusterSnapshots Rotverschiebung: BatchModify ClusterSnapshots Rotverschiebung: CancelResize redshift: Schnappschuss CopyCluster redshift: Profil CreateAuthentication redshift: CreateCluster Rotverschiebung: CreateCluster ParameterGroup Rotverschiebung: CreateCluster SecurityGroup redshift: Schnappschuss CreateCluster Rotverschiebung: CreateCluster SubnetGroup Rotverschiebung: CreateCustom DomainAssociation redshift: Zugriff CreateEndpoint redshift: Abonnement CreateEvent redshift: CreateHsm ClientCertificate

Servicepräfix	Aktionen
	redshift: Konfiguration CreateHsm Redshift: CreateRedshift IdcApplication Redshift: Aktion CreateScheduled Rotverschiebung: CreateSnapshot CopyGrant redshift: Zeitplan CreateSnapshot redshift: Limit CreateUsage redshift: Teilen DeauthorizeData redshift: Profil DeleteAuthentication redshift: DeleteCluster Rotverschiebung: DeleteCluster ParameterGroup Rotverschiebung: DeleteCluster SecurityGroup redshift: Schnappschuss DeleteCluster Rotverschiebung: DeleteCluster SubnetGroup Rotverschiebung: DeleteCustom DomainAssociation redshift: Zugriff DeleteEndpoint redshift: Abonnement DeleteEvent redshift: DeleteHsm ClientCertificate redshift: Konfiguration DeleteHsm Redshift: DeletePartner Redshift: Aktion DeleteScheduled Rotverschiebung: DeleteSnapshot CopyGrant

Servicepräfix	Aktionen
	redshift: Zeitplan DeleteSnapshot redshift: Limit DeleteUsage redshift: Attribute DescribeAccount redshift: Profile DescribeAuthentication Redshift: DescribeCluster DbRevisions Rotverschiebung: DescribeCluster ParameterGroups Redshift: Parameter DescribeCluster Rotverschiebung: DescribeClusters Rotverschiebung: DescribeCluster SecurityGroups redshift: Schnappschüsse DescribeCluster Rotverschiebung: DescribeCluster SubnetGroups redshift: Spuren DescribeCluster redshift: Versionen DescribeCluster Redshift: DescribeCustom DomainAssociations redshift: Aktien DescribeData redshift: Verbraucher DescribeData SharesFor redshift: Produzent DescribeData SharesFor redshift: DescribeDefault ClusterParameters redshift: Zugriff DescribeEndpoint redshift: Autorisierung DescribeEndpoint redshift: Kategorien DescribeEvent

Servicepräfix	Aktionen
	Rotverschiebung: DescribeEvents redshift: Abonnements DescribeEvent redshift: DescribeHsm ClientCertificates redshift: Konfigurationen DescribeHsm redshift: Integrationen DescribeInbound redshift: Status DescribeLogging Rotverschiebung: DescribeNode ConfigurationOptions Rotverschiebung: DescribeOrderable ClusterOptions Rotverschiebung: DescribePartners Rotverschiebung: DescribeRedshift IdcApplications Rotverschiebung: Status DescribeReserved NodeExchange Rotverschiebung: DescribeReserved NodeOfferings Redshift: Knoten DescribeReserved Rotverschiebung: DescribeResize redshift: Aktionen DescribeScheduled Redshift: DescribeSnapshot CopyGrants redshift: Zeitpläne DescribeSnapshot Redshift: DescribeStorage Rotverschiebung: DescribeTable RestoreStatus redshift: Grenzwerte DescribeUsage Rotverschiebung: DisableLogging

Servicepräfix	Aktionen
	redshift: Kopieren DisableSnapshot Rotverschiebung: DisassociateData ShareConsumer Rotverschiebung: EnableLogging redshift: Kopieren EnableSnapshot redshift: Berechne FailoverPrimary redshift: Anmeldeinformationen GetCluster redshift: IAM GetCluster CredentialsWith Rotverschiebung: GetReserved NodeExchange ConfigurationOptions redshift: Angebote GetReserved NodeExchange redshift: ListRecommendations redshift: Konfiguration ModifyAqua redshift: Profil ModifyAuthentication redshift: ModifyCluster Rotverschiebung: ModifyCluster DbRevision Rotverschiebung: ModifyCluster IamRoles redshift: Wartung ModifyCluster Redshift: ModifyCluster ParameterGroup redshift: Schnappschuss ModifyCluster Rotverschiebung: ModifyCluster SnapshotSchedule Rotverschiebung: ModifyCluster SubnetGroup

Servicepräfix	Aktionen
	Rotverschiebung: ModifyCustom DomainAssociation redshift: Zugriff ModifyEndpoint redshift: Abonnement ModifyEvent redshift: Aktion ModifyScheduled Redshift: Zeitraum ModifySnapshot CopyRetention redshift: Zeitplan ModifySnapshot redshift: Limit ModifyUsage Rotverschiebung: PauseCluster Rotverschiebung: PurchaseReserved NodeOffering Rotverschiebung: RebootCluster redshift: Teilen RejectData redshift: ResetCluster ParameterGroup Rotverschiebung: ResizeCluster Rotverschiebung: RestoreFrom ClusterSnapshot redshift: Schnappschuss RestoreTable FromCluster Rotverschiebung: ResumeCluster Rotverschiebung: Eintritt RevokeCluster SecurityGroup redshift: Zugriff RevokeEndpoint redshift: Zugriff RevokeSnapshot redshift: Schlüssel RotateEncryption redshift: Status UpdatePartner

Servicepräfix	Aktionen
redshift-data	redshift-data: Aussage BatchExecute
	redshift-data: CancelStatement
	Redshift-Daten: DescribeStatement
	Redshift-Daten: DescribeTable
	Redshift-Daten: ExecuteStatement
	redshift-data: Ergebnis GetStatement
	Redshift-Daten: ListDatabases
	Redshift-Daten: ListSchemas
	Redshift-Daten: ListStatements
	Redshift-Daten: ListTables

Servicepräfix	Aktionen
refactor-spaces	Refaktor-Räume: CreateApplication Refaktor-Räume: CreateEnvironment Refaktor-Räume: CreateRoute Refaktor-Räume: CreateService Refaktor-Räume: DeleteApplication Refaktor-Räume: DeleteEnvironment Refactor-Spaces: Richtlinie DeleteResource Refactor-Spaces: DeleteRoute Refaktor-Räume: DeleteService Refaktor-Räume: GetApplication Refaktor-Räume: GetEnvironment Refactor-Spaces: Richtlinie GetResource Refactor-Spaces: GetRoute Refaktor-Räume: GetService Refaktor-Räume: ListApplications Refaktor-Räume: ListEnvironments Refaktor-Räume: Vpcs ListEnvironment Refaktor-Räume: ListRoutes Refaktor-Räume: ListServices Refactor-Spaces: Richtlinie PutResource Refactor-Spaces: UpdateRoute

Servicepräfix	Aktionen
Rekognition	Anerkennung: AssociateFaces
	Wiedererkennung: CompareFaces
	Rekognition: Ausführung CopyProject
	Wiedererkennung: CreateCollection
	Wiedererkennung: CreateDataset
	Wiedererkennung: CreateFace LivenessSession
	Wiedererkennung: CreateProject
	Rekognition: Ausführung CreateProject
	Rekognition: Prozessor CreateStream
	Wiedererkennung: CreateUser
	Wiedererkennung: DeleteCollection
	Wiedererkennung: DeleteDataset
	Wiedererkennung: DeleteFaces
	Wiedererkennung: DeleteProject
	Rekognition: Richtlinie DeleteProject
	Rekognition: Version DeleteProject
	Rekognition: Prozessor DeleteStream
	Wiedererkennung: DeleteUser
	Wiedererkennung: DescribeCollection
	Wiedererkennung: DescribeDataset
	Wiedererkennung: DescribeProjects

Servicepräfix	Aktionen
	Rekognition: Versionen DescribeProject
	Rekognition: Prozessor DescribeStream
	Rekognition: Etiketten DetectCustom
	Rekognition: DetectFaces
	Wiedererkennung: DetectLabels
	Rekognition: Etiketten DetectModeration
	Rekognition: Ausrüstung DetectProtective
	Rekognition: DetectText
	Wiedererkennung: DisassociateFaces
	Rekognition: Einträge DistributeDataset
	Rekognition: Informationen GetCelebrity
	Rekognition: Anerkennung GetCelebrity
	Rekognition: Mäßigung GetContent
	Rekognition: Erkennung GetFace
	Rekognition: Ergebnisse GetFace LivenessSession
	Rekognition: Suche GetFace
	Rekognition: Erkennung GetLabel
	Wiedererkennung: GetMedia AnalysisJob
	Rekognition: Nachverfolgung GetPerson
	Rekognition: Erkennung GetSegment
	Rekognition: Erkennung GetText

Servicepräfix	Aktionen
	Wiedererkennung: IndexFaces
	Wiedererkennung: ListCollections
	Rekognition: Einträge ListDataset
	Rekognition: Etiketten ListDataset
	Rekognition: ListFaces
	Wiedererkennung: ListMedia AnalysisJobs
	Rekognition: Richtlinien ListProject
	Rekognition: Prozessoren ListStream
	Wiedererkennung: ListUsers
	Rekognition: Richtlinie PutProject
	Anerkennung: RecognizeCelebrities
	Wiedererkennung: SearchFaces
	Wiedererkennung: SearchFaces ByImage
	Wiedererkennung: SearchUsers
	Wiedererkennung: SearchUsers ByImage
	Rekognition: Anerkennung StartCelebrity
	Rekognition: Mäßigung StartContent
	Rekognition: Erkennung StartFace
	Wiedererkennung: StartFace LivenessSession
	Rekognition: Suche StartFace
	Rekognition: Erkennung StartLabel

Servicepräfix	Aktionen
	Wiedererkennung: StartMedia AnalysisJob
	Rekognition: Nachverfolgung StartPerson
	Rekognition: Ausführung StartProject
	Rekognition: Erkennung StartSegment
	Rekognition: Prozessor StartStream
	Rekognition: Erkennung StartText
	Rekognition: Ausführung StopProject
	Rekognition: Prozessor StopStream
	Rekognition: Einträge UpdateDataset
	Rekognition: Prozessor UpdateStream

Servicepräfix	Aktionen
resiliencehub	Resilienz-Hub: AddDraft AppVersion ResourceMappings Resiliencehub: CreateApp resiliencehub: Komponente CreateApp VersionApp Resiliencehub: CreateApp VersionResource resiliencehub: Vorlage CreateRecommendation resiliencehub: Richtlinie CreateResiliency Resiliencehub: DeleteApp resiliencehub: Bewertung DeleteApp Resiliencehub: DeleteApp InputSource resiliencehub: Komponente DeleteApp VersionApp Resiliencehub: DeleteApp VersionResource resiliencehub: Vorlage DeleteRecommendation resiliencehub: Richtlinie DeleteResiliency Resiliencehub: DescribeApp resiliencehub: Bewertung DescribeApp resiliencehub: Version DescribeApp resiliencehub: Komponente DescribeApp VersionApp Resiliencehub: DescribeApp VersionResource Resiliencehub: DescribeApp VersionResources ResolutionStatus Resiliencehub: DescribeApp VersionTemplate resiliencehub: Status DescribeDraft AppVersion ResourcesImport

Servicepräfix	Aktionen
	resiliencehub: Richtlinie DescribeResiliency Resiliencehub: ImportResources ToDraft AppVersion Resiliencehub: Empfehlungen ListAlarm Resiliencehub: Bewertungen ListApp Resiliencehub: ListApp ComponentCompliances Resiliencehub: ListApp ComponentRecommendations Resiliencehub: ListApp InputSources Resilienzzentrum: ListApps resiliencehub: Komponenten ListApp VersionApp resiliencehub: Zuordnungen ListApp VersionResource resiliencehub: ListApp VersionResources resiliencehub: Versionen ListApp resiliencehub: Vorlagen ListRecommendation resiliencehub: Richtlinien ListResiliency Resiliencehub: Empfehlungen ListSop Resiliencehub: ListSuggested ResiliencyPolicies Resiliencehub: Empfehlungen ListTest Resiliencehub: Ressourcen ListUnsupported AppVersion resiliencehub: Version PublishApp resiliencehub: Vorlage PutDraft AppVersion Resiliencehub: RemoveDraft AppVersion ResourceMappings

Servicepräfix	Aktionen
	Resiliencehub: ResolveApp VersionResources resiliencehub: Bewertung StartApp Resiliencehub: UpdateApp resiliencehub: Ausführung UpdateApp resiliencehub: Komponente UpdateApp VersionApp Resiliencehub: UpdateApp VersionResource resiliencehub: Richtlinie UpdateResiliency

Servicepräfix	Aktionen
resource-explorer-2	resource-explorer-2: Ansicht AssociateDefault
	resource-explorer-2: BatchGet Ansicht
	resource-explorer-2: CreateIndex
	Ressourcen-Explorer-2: CreateView
	Ressourcen-Explorer-2: DeleteIndex
	Ressourcen-Explorer-2: DeleteView
	resource-explorer-2: Ansicht DisassociateDefault
	resource-explorer-2: GetAccount LevelService Konfiguration
	resource-explorer-2: GetDefault Ansicht
	resource-explorer-2: GetIndex
	Ressourcen-Explorer-2: ListIndexes
	Ressourcen-Explorer-2: ListIndexes ForMembers
	Ressourcen-Explorer-2: ListSupported ResourceTypes
	Ressourcen-Explorer-2: ListViews
	resource-explorer-2:Search
	resource-explorer-2: Typ UpdateIndex
	resource-explorer-2: UpdateView

Servicepräfix	Aktionen
resource-groups	Ressourcengruppen: CreateGroup Ressourcengruppen: DeleteGroup Ressourcengruppen: Einstellungen GetAccount Ressourcengruppen: GetGroup Ressourcengruppen: Konfiguration GetGroup Ressourcengruppen: Abfrage GetGroup Ressourcengruppen: GroupResources Ressourcengruppen: Ressourcen ListGroup Ressourcengruppen: ListGroups Ressourcengruppen: Konfiguration PutGroup Ressourcengruppen: SearchResources Ressourcengruppen: UngroupResources Ressourcengruppen: Einstellungen UpdateAccount Ressourcengruppen: UpdateGroup Ressourcengruppen: Abfrage UpdateGroup

Servicepräfix	Aktionen
robomaker	Robomaker: Welten BatchDelete
	Robomaker: BatchDescribe SimulationJob
	Robomaker: Job CancelDeployment
	Robomaker: Job CancelSimulation
	Robomaker: CancelSimulation JobBatch
	Roboterhersteller: CancelWorld ExportJob
	Roboterhersteller: CancelWorld GenerationJob
	Robomaker: Job CreateDeployment
	Robomaker: CreateFleet
	Roboterhersteller: CreateRobot
	robomaker: Anwendung CreateRobot
	Robomaker: CreateRobot ApplicationVersion
	robomaker: Anwendung CreateSimulation
	Robomaker: CreateSimulation ApplicationVersion
	Robomaker: Job CreateSimulation
	Robomaker: CreateWorld ExportJob
	Roboterhersteller: CreateWorld GenerationJob
	robomaker: Vorlage CreateWorld
	Robomaker: DeleteFleet
	Roboterhersteller: DeleteRobot
	robomaker: Anwendung DeleteRobot

Servicepräfix	Aktionen
	robomaker: Anwendung DeleteSimulation
	robomaker: Vorlage DeleteWorld
	Robomaker: DeregisterRobot
	Robomaker: Job DescribeDeployment
	Robomaker: DescribeFleet
	Roboterhersteller: DescribeRobot
	robomaker: Anwendung DescribeRobot
	robomaker: Anwendung DescribeSimulation
	Robomaker: Job DescribeSimulation
	Robomaker: DescribeSimulation JobBatch
	Roboterhersteller: DescribeWorld
	Roboterhersteller: DescribeWorld ExportJob
	Roboterhersteller: DescribeWorld GenerationJob
	robomaker: Vorlage DescribeWorld
	Robomaker: GetWorld TemplateBody
	Robomaker: Stellenangebote ListDeployment
	Robomaker: ListFleets
	robomaker: Anwendungen ListRobot
	Robomaker: ListRobots
	robomaker: Anwendungen ListSimulation
	Robomaker: ListSimulation JobBatches

Servicepräfix	Aktionen
	Robomaker: Stellenangebote ListSimulation
	Robomaker: ListWorld ExportJobs
	Roboterhersteller: ListWorld GenerationJobs
	Roboterhersteller: ListWorlds
	robomaker: Vorlagen ListWorld
	Robomaker: RegisterRobot
	Robomaker: Job RestartSimulation
	Robomaker: StartSimulation JobBatch
	Robomaker: Job SyncDeployment
	robomaker: Bewerbung UpdateRobot
	robomaker: Anwendung UpdateSimulation
	robomaker: Vorlage UpdateWorld

Servicepräfix	Aktionen
rolesanywhere	Rollen überall: CreateProfile rolesanywhere: Anker CreateTrust rolesanywhere: Zuordnung DeleteAttribute rolesanywhere: DeleteCrl Rollen überall: DeleteProfile rolesanywhere: Anker DeleteTrust rolesanywhere: DisableCrl Rollen überall: DisableProfile rolesanywhere: Anker DisableTrust rolesanywhere: EnableCrl Rollen überall: EnableProfile rolesanywhere: Anker EnableTrust rolesanywhere: GetCrl Rollen überall: GetProfile Rollen überall: GetSubject rolesanywhere: Anker GetTrust rolesanywhere: ImportCrl Rollen überall: ListCrls Rollen überall: ListProfiles Rollen überall: ListSubjects rolesanywhere: Anker ListTrust

Servicepräfix	Aktionen
	rolesanywhere: Zuordnung PutAttribute rolesanywhere: Einstellungen PutNotification rolesanywhere: Einstellungen ResetNotification rolesanywhere: UpdateCrl Rollen überall: UpdateProfile rolesanywhere: Anker UpdateTrust

Servicepräfix	Aktionen
route53	Route 53: ActivateKey SigningKey
	Route 53: VPC-Zone zuordnen WithHosted
	route53: Sammlung ChangeCidr
	route53: ChangeResource RecordSets
	route53: Sammlung CreateCidr
	route53: Prüfen CreateHealth
	route53: Zone CreateHosted
	Route 53: CreateKey SigningKey
	Route 53: CreateQuery LoggingConfig
	Route 53: CreateReusable DelegationSet
	route53: Richtlinie CreateTraffic
	Route53: CreateTraffic PolicyInstance
	Route 53: CreateTraffic PolicyVersion
	Route 53: VPC erstellen AssociationAuthorization
	Route 53: DeactivateKey SigningKey
	route53: Sammlung DeleteCidr
	route53: Prüfen DeleteHealth
	route53: Zone DeleteHosted
	Route 53: DeleteKey SigningKey
	Route 53: DeleteQuery LoggingConfig
	Route 53: DeleteReusable DelegationSet

Servicepräfix	Aktionen
	route53: Richtlinie DeleteTraffic
	Route53: DeleteTraffic PolicyInstance
	Route 53: VPC löschen AssociationAuthorization
	Route 53: Zone DNSSEC DisableHosted
	Route 53FromHosted: VPC-Zone trennen
	Route 53: Zone DNSSEC EnableHosted
	route53GetAccount: Grenze
	Route53: GetChange
	Route 53: GetChecker IpRanges
	route53:GetDNSSEC
	route53: Standort GetGeo
	route53: Überprüfen GetHealth
	Route53: GetHealth CheckCount
	Route 53: GetHealth CheckLast FailureReason
	Route 53: GetHealth CheckStatus
	Route53: Zone GetHosted
	Route 53: GetHosted ZoneCount
	Route 53: GetHosted ZoneLimit
	Route 53: GetQuery LoggingConfig
	Route 53: GetReusable DelegationSet
	route53: Grenze GetReusable DelegationSet

Servicepräfix	Aktionen
	route53: Richtlinie GetTraffic
	Route53: GetTraffic PolicyInstance
	route53: Anzahl GetTraffic PolicyInstance
	route53: Blöcke ListCidr
	route53: Sammlungen ListCidr
	route53: Standorte ListCidr
	route53: Standorte ListGeo
	route53: Schecks ListHealth
	route53: Zonen ListHosted
	route53: Name ListHosted ZonesBy
	Route 53: VPC ListHosted ZonesBy
	Route 53: ListQuery LoggingConfigs
	Route 53: ListResource RecordSets
	Route 53: ListReusable DelegationSets
	route53: Richtlinien ListTraffic
	route53: ListTraffic PolicyInstances
	Route53: Zone ListTraffic PolicyInstances ByHosted
	Route 53: ListTraffic PolicyInstances ByPolicy
	Route 53: ListTraffic PolicyVersions
	Route 53: VPC auflisten AssociationAuthorizations
	route53:TestDNSAnswer

Servicepräfix	Aktionen
	route53: Überprüfen UpdateHealth Route53: UpdateHosted ZoneComment Route 53: UpdateTraffic PolicyComment Route 53: UpdateTraffic PolicyInstance

Servicepräfix	Aktionen
route53-recovery-control-config	route53-recovery-control-config: CreateCluster route53-recovery-control-config: Bedienfeld CreateControl route53-recovery-control-config: Steuerung CreateRouting route53-recovery-control-config: Regel CreateSafety route53-recovery-control-config: DeleteCluster route53-recovery-control-config: Bedienfeld DeleteControl route53-recovery-control-config: Steuerung DeleteRouting route53-recovery-control-config: Regel DeleteSafety route53-recovery-control-config: DescribeCluster route53-recovery-control-config: Bedienfeld DescribeControl route53-recovery-control-config: Steuerung DescribeRouting route53-recovery-control-config: Regel DescribeSafety route53-recovery-control-config: Richtlinie GetResource route53-recovery-control-config: Route53 ListAssociated HealthChecks route53-recovery-control-config: ListClusters route53-recovery-control-config: Bedienfelder ListControl route53-recovery-control-config: Steuerungen ListRouting route53-recovery-control-config: Regeln ListSafety route53-recovery-control-config: Bedienfeld UpdateControl route53-recovery-control-config: Steuerung UpdateRouting

Servicepräfix	Aktionen
	route53-recovery-control-config: Regel UpdateSafety

Servicepräfix	Aktionen
route53-recovery-readiness	Route53-Recovery-Bereitschaft: CreateCell Route53-Wiederherstellungsbereitschaft: CreateCross AccountAuthorization route53-recovery-ready: Überprüfen CreateReadiness route53-recovery-ready: Gruppe CreateRecovery route53-recovery-ready: Eingestellt CreateResource route53-Recovery-Readiness: DeleteCell Route53-Wiederherstellungsbereitschaft: DeleteCross AccountAuthorization route53-recovery-ready: Überprüfen DeleteReadiness route53-recovery-ready: Gruppe DeleteRecovery route53-recovery-ready: Eingestellt DeleteResource route53-recovery-ready: Empfehlungen GetArchitecture Route53-Recovery-Bereitschaft: GetCell Route53-Wiederherstellungsbereitschaft: GetCell Readiness Summary route53-recovery-ready: Überprüfen GetReadiness route53-recovery-ready: Status GetReadiness CheckResource Route53-Wiederherstellungsbereitschaft: GetReadiness CheckStatus route53-recovery-ready: Gruppe GetRecovery route53-recovery-ready: Zusammenfassung GetRecovery GroupReadiness

Servicepräfix	Aktionen
	route53-recovery-ready: Eingestellt GetResource
	route53-Recovery-Readiness: ListCells
	Route53-Wiederherstellungsbereitschaft: ListCross AccountAuthorizations
	route53-recovery-ready: Prüft ListReadiness
	route53-recovery-ready: Gruppen ListRecovery
	route53-recovery-ready: Sätze ListResource
	route53-Recovery-Bereitschaft: ListRules
	Route53-Wiederherstellungsbereitschaft: UpdateCell
	route53-recovery-ready: Überprüfen UpdateReadiness
	route53-recovery-ready: Gruppe UpdateRecovery
	route53-recovery-ready: Eingestellt UpdateResource

Servicepräfix	Aktionen
route53resolver	Route53-Resolver: AssociateFirewall RuleGroup route53resolver: Adresse AssociateResolver EndpointIp route53resolver: Config AssociateResolver QueryLog route53resolver: AssociateResolver Regel route53resolver: CreateFirewall DomainList route53resolver: Regel CreateFirewall route53resolver: CreateFirewall RuleGroup route53resolver: Endpunkt CreateResolver route53resolver: Config CreateResolver QueryLog route53resolver: CreateResolver Regel route53resolver: DeleteFirewall DomainList route53resolver: Regel DeleteFirewall route53resolver: DeleteFirewall RuleGroup route53resolver: DeleteOutpost Resolver route53resolver: DeleteResolver Endpunkt route53resolver: Config DeleteResolver QueryLog route53resolver: DeleteResolver Regel route53resolver: DisassociateFirewall RuleGroup route53resolver: Adresse DisassociateResolver EndpointIp route53resolver: Config DisassociateResolver QueryLog route53resolver: DisassociateResolver Regel

Servicepräfix	Aktionen
	route53resolver: Config GetFirewall Route53-Resolver: GetFirewall DomainList Route53-Resolver: GetFirewall RuleGroup route53resolver: Assoziation GetFirewall RuleGroup route53resolver: GetFirewall RuleGroup Richtlinie route53resolver: GetOutpost Resolver route53resolver: Config GetResolver Route53-Resolver: GetResolver DnssecConfig route53resolver: Endpunkt GetResolver route53resolver: Config GetResolver QueryLog Route53-Resolver: GetResolver QueryLog ConfigAssociation Route53-Resolver: GetResolver QueryLog ConfigPolicy route53resolver: Regel GetResolver route53resolver: GetResolver RuleAssociation Route53-Resolver: GetResolver RulePolicy route53resolver: Domänen ImportFirewall route53resolver: ListFirewall Konfigurationen Route53-Resolver: ListFirewall DomainLists route53resolver: Domänen ListFirewall route53resolver: ListFirewall RuleGroup Assoziationen route53resolver: ListFirewall RuleGroups

Servicepräfix	Aktionen
	route53resolver: Regeln ListFirewall route53resolver: ListOutpost Resolver route53resolver: ListResolver Konfigurationen Route53-Resolver: ListResolver DnssecConfigs route53resolver: Adressen ListResolver EndpointIp route53resolver: ListResolver Endpunkte Route53-Resolver: ListResolver QueryLog ConfigAssociations route53resolver: Konfigurationen ListResolver QueryLog Route53-Resolver: ListResolver RuleAssociations route53resolver: Regeln ListResolver route53resolver: PutFirewall RuleGroup Richtlinie route53resolver: PutResolver QueryLog ConfigPolicy route53resolver: Config UpdateFirewall route53resolver: UpdateFirewall Domänen route53resolver: UpdateFirewall Regel route53resolver: UpdateFirewall RuleGroup Assoziation route53resolver: UpdateOutpost Resolver route53resolver: Config UpdateResolver Route53-Resolver: UpdateResolver DnssecConfig route53resolver: Endpunkt UpdateResolver route53resolver: UpdateResolver Regel

Servicepräfix	Aktionen
rum	rum: BatchCreate RumMetric Definitionen Rum: BatchDelete RumMetric Definitionen Rum: BatchGet RumMetric Definitionen rum: CreateApp Überwachen rum: DeleteApp Überwachen Rum: DeleteRum MetricsDestination Rum: GetApp Überwachen Rum: GetApp MonitorData Rum: ListApp Monitore Rum: ListRum MetricsDestinations Rum: PutRum MetricsDestination Rum: UpdateApp Überwachen Rum: UpdateRum MetricDefinition

Servicepräfix	Aktionen
S3	s3: AssociateAccess GrantsIdentity Zentrum s3: CreateAccess Zuschuss s3: CreateAccess GrantsInstance s3: CreateAccess GrantsLocation s3: CreateAccess Punkt s3: CreateAccess PointFor ObjectLambda s3: CreateBucket s3: CreateJob s3: CreateMulti RegionAccess Punkt s3: DeleteAccess Zuschuss s3: DeleteAccess GrantsInstance s3: DeleteAccess GrantsInstance ResourcePolicy s3: DeleteAccess GrantsLocation s3: DeleteAccess Punkt s3: DeleteAccess PointFor ObjectLambda s3: DeleteAccess PointPolicy s3: DeleteAccess PointPolicy ForObject Lambda s3: Sperren PutAccount PublicAccess s3: DeleteBucket s3: PutAnalytics Konfiguration s3: PutBucket CORS

Servicepräfix	Aktionen
	s3: Konfiguration PutEncryption s3: PutIntelligent TieringConfiguration s3: PutInventory Konfiguration s3: PutLifecycle Konfiguration s3: PutMetrics Konfiguration s3: PutBucket OwnershipControls s3: DeleteBucket Richtlinie s3: PutBucket PublicAccess Blockieren s3: PutReplication Konfiguration s3: DeleteBucket Webseite s3: DeleteMulti RegionAccess Punkt s3: DeleteStorage LensConfiguration s3: DescribeJob s3: DescribeMulti RegionAccess PointOperation s3: DissociateAccess GrantsIdentity Zentrum s3: GetAccelerate Konfiguration s3: GetAccess Zuschuss s3: GetAccess GrantsInstance s3: GetAccess GrantsInstance ForPrefix s3: GetAccess GrantsInstance ResourcePolicy s3: GetAccess GrantsLocation

Servicepräfix	Aktionen
	s3: GetAccess Punkt s3: GetAccess PointConfiguration ForObject Lambda s3: GetAccess PointFor ObjectLambda s3: GetAccess PointPolicy s3: GetAccess PointPolicy ForObject Lambda s3: Status GetAccess PointPolicy s3: GetAccess PointPolicy StatusFor ObjectLambda s3: GetAccount PublicAccess Sperren s3: GetBucket Acl s3: GetAnalytics Konfiguration s3: GetBucket CORS s3: Konfiguration GetEncryption s3: GetIntelligent TieringConfiguration s3: GetInventory Konfiguration s3: GetLifecycle Konfiguration s3: GetBucket Standort s3: GetBucket Protokollierung s3: GetMetrics Konfiguration s3: GetBucket Benachrichtigung s3: GetBucket ObjectLock Konfiguration s3: GetBucket OwnershipControls

Servicepräfix	Aktionen
	s3: GetBucket Richtlinie s3: GetBucket PolicyStatus s3: GetBucket PublicAccess Sperren s3: GetReplication Konfiguration s3: GetBucket RequestPayment s3: GetBucket Versionierung s3: Webseite GetBucket s3: GetData Zugriff s3: GetMulti RegionAccess Punkt s3: GetMulti RegionAccess PointPolicy s3: GetMulti RegionAccess PointPolicy Status s3: GetMulti RegionAccess PointRoutes s3: GetObject Attribute s3: GetStorage LensConfiguration s3: GetStorage LensDashboard s3: ListAccess Zuschüsse s3: ListAccess GrantsInstances s3: ListAccess GrantsLocations s3: ListAccess Punkte s3: ListAccess PointsFor ObjectLambda s3: ListAll MyBuckets

Servicepräfix	Aktionen
	s3: ListJobs s3: ListBucket MultipartUploads s3: ListMulti RegionAccess Punkte s3: ListStorage LensConfigurations s3: PutAccelerate Konfiguration s3: PutAccess GrantsInstance ResourcePolicy s3: PutAccess PointConfiguration ForObject Lambda s3: PutAccess PointPolicy s3: PutAccess PointPolicy ForObject Lambda s3: Sperren PutAccount PublicAccess s3: PutBucket Acl s3: PutAnalytics Konfiguration s3: PutBucket CORS s3: Konfiguration PutEncryption s3: PutIntelligent TieringConfiguration s3: PutInventory Konfiguration s3: PutLifecycle Konfiguration s3: PutBucket Protokollierung s3: PutMetrics Konfiguration s3: PutBucket Benachrichtigung s3: PutBucket ObjectLock Konfiguration

Servicepräfix	Aktionen
	s3: PutBucket OwnershipControls s3: PutBucket Richtlinie s3: PutBucket PublicAccess Blockieren s3: PutReplication Konfiguration s3: PutBucket RequestPayment s3: PutBucket Versionierung s3: Webseite PutBucket s3: PutMulti RegionAccess PointPolicy s3: PutStorage LensConfiguration s3: SubmitMulti RegionAccess PointRoutes s3: UpdateAccess GrantsLocation s3: UpdateJob Priorität s3: UpdateJob Status
s3-outposts	s3-Außenposten: CreateEndpoint s3-Außenposten: DeleteEndpoint s3-Außenposten: ListEndpoints s3-Außenposten: mit S3 ListOutposts s3-Outposts: ListShared Endpunkte

Servicepräfix	Aktionen
sagemaker-geospatial	Sagemaker-Geospatial: DeleteEarth ObservationJob Sagemaker-Geospatial: DeleteVector EnrichmentJob Sagemaker-Geospatial: ExportEarth ObservationJob Sagemaker-Geospatial: ExportVector EnrichmentJob Sagemaker-Geospatial: GetEarth ObservationJob Sagemaker-Geospatial: GetRaster DataCollection Sagemaker-Geospatial: GetTile Sagemaker-Geospatial: GetVector EnrichmentJob Sagemaker-Geospatial: ListEarth ObservationJobs Sagemaker-Geospatial: ListRaster DataCollections Sagemaker-Geospatial: ListVector EnrichmentJobs Sagemaker-Geospatial: SearchRaster DataCollection Sagemaker-Geospatial: StartEarth ObservationJob Sagemaker-Geospatial: StartVector EnrichmentJob Sagemaker-Geospatial: StopEarth ObservationJob Sagemaker-Geospatial: StopVector EnrichmentJob

Servicepräfix	Aktionen
savingsplans	Sparpläne: Plan CreateSavings
	Sparpläne: DeleteQueued SavingsPlan
	Sparpläne: DescribeSavings PlanRates
	Sparpläne: Pläne DescribeSavings
	Sparpläne: Tarife DescribeSavings PlansOffering
	Sparpläne: DescribeSavings PlansOfferings
	Sparpläne: Plan ReturnSavings

Servicepräfix	Aktionen
schemas	Schemas: CreateDiscoverer Schemas: CreateRegistry Schemas: CreateSchema Schemas: DeleteDiscoverer Schemas: DeleteRegistry Schemas: Richtlinie DeleteResource Schemas: DeleteSchema Schemas: Version DeleteSchema Schemas: Bindung DescribeCode Schemas: DescribeDiscoverer Schemas: DescribeRegistry Schemas: DescribeSchema Schemas: ExportSchema Schemas: GetCode BindingSource Schemas: Schema GetDiscovered Schemas: Richtlinie GetResource Schemas: ListDiscoverers Schemas: ListRegistries Schemas: ListSchemas Schemas: Versionen ListSchema Schemas: Bindung PutCode

Servicepräfix	Aktionen
	Schemas: Richtlinie PutResource Schemas: SearchSchemas Schemas: StartDiscoverer Schemas: StopDiscoverer Schemas: UpdateDiscoverer Schemas: UpdateRegistry Schemas: UpdateSchema
sdb	sdb: CreateDomain sdb: DeleteDomain sdb: DomainMetadata sdb: ListDomains

Servicepräfix	Aktionen
secretsmanager	secretsmanager: Geheim CancelRotate secretsmanager: CreateSecret secretsmanager: Richtlinie DeleteResource secretsmanager: DeleteSecret Verwalter von Geheimnissen: DescribeSecret secretsmanager: Passwort GetRandom secretsmanager: Richtlinie GetResource secretsmanager: Wert GetSecret secretsmanager: ListSecrets Verwalter von Geheimnissen: ListSecret VersionIds secretsmanager: Richtlinie PutResource secretsmanager: Wert PutSecret secretsmanager: RemoveRegions FromReplication Verwalter von Geheimnissen: ReplicateSecret ToRegions Verwalter von Geheimnissen: RestoreSecret Verwalter von Geheimnissen: RotateSecret Verwalter von Geheimnissen: StopReplication ToReplica Verwalter von Geheimnissen: UpdateSecret secretsmanager: Richtlinie ValidateResource

Servicepräfix	Aktionen
securityhub	securityhub: Einladung AcceptAdministrator
	Securityhub: AcceptInvitation
	Sicherheitszentrum: BatchDelete AutomationRules
	securityhub: Standards BatchDisable
	securityhub: Standards BatchEnable
	Securityhub: BatchGet AutomationRules
	securityhub: Verbände BatchGet ConfigurationPolicy
	Securityhub: BatchGet SecurityControls
	securityhub: Verbände BatchGet StandardsControl
	securityhub: Ergebnisse BatchImport
	Securityhub: BatchUpdate AutomationRules
	securityhub: Ergebnisse BatchUpdate
	securityhub: Verbände BatchUpdate StandardsControl
	securityhub: Ziel CreateAction
	securityhub: Regel CreateAutomation
	securityhub: Richtlinie CreateConfiguration
	securityhub: Aggregator CreateFinding
	Sicherheitshub: CreateInsight
	Sicherheitszentrum: CreateMembers
	Sicherheitszentrum: DeclineInvitations
	securityhub: Ziel DeleteAction

Servicepräfix	Aktionen
	securityhub: Richtlinie DeleteConfiguration
	securityhub: Aggregator DeleteFinding
	Sicherheitshub: DeleteInsight
	Sicherheitszentrum: DeleteInvitations
	Sicherheitszentrum: DeleteMembers
	securityhub: Ziele DescribeAction
	Securityhub: DescribeHub
	securityhub: Konfiguration DescribeOrganization
	Securityhub: DescribeProducts
	Sicherheitszentrum: DescribeStandards
	securityhub: Produkt DisableImport FindingsFor
	Securityhub: DisableOrganization AdminAccount
	Securityhub: Knotenpunkt DisableSecurity
	Sicherheitszentrum: DisassociateFrom AdministratorAccount
	Sicherheitszentrum: DisassociateFrom MasterAccount
	Sicherheitszentrum: DisassociateMembers
	securityhub: Produkt EnableImport FindingsFor
	Securityhub: EnableOrganization AdminAccount
	Securityhub: Knotenpunkt EnableSecurity
	securityhub: Konto GetAdministrator
	securityhub: Richtlinie GetConfiguration

Servicepräfix	Aktionen
	Securityhub: GetConfiguration PolicyAssociation
	securityhub: Standards GetEnabled
	securityhub: Aggregator GetFinding
	securityhub: Geschichte GetFinding
	Securityhub: GetFindings
	securityhub: Ergebnisse GetInsight
	Securityhub: GetInsights
	securityhub: Anzahl GetInvitations
	securityhub: Konto GetMaster
	securityhub: GetMembers
	Sicherheitszentrum: GetSecurity ControlDefinition
	Sicherheitszentrum: InviteMembers
	securityhub: Regeln ListAutomation
	securityhub: Richtlinien ListConfiguration
	Securityhub: ListConfiguration PolicyAssociations
	securityhub: Importieren ListEnabled ProductsFor
	securityhub: Aggregatoren ListFinding
	Securityhub: ListInvitations
	Sicherheitszentrum: ListMembers
	Sicherheitszentrum: ListOrganization AdminAccounts
	Sicherheitszentrum: ListSecurity ControlDefinitions

Servicepräfix	Aktionen
	Sicherheitszentrum: ListStandards ControlAssociations
	Sicherheitszentrum: StartConfiguration PolicyAssociation
	Sicherheitszentrum: StartConfiguration PolicyDisassociation
	securityhub: Ziel UpdateAction
	securityhub: Richtlinie UpdateConfiguration
	securityhub: Aggregator UpdateFinding
	Sicherheitshub: UpdateFindings
	Sicherheitszentrum: UpdateInsight
	securityhub: Konfiguration UpdateOrganization
	securityhub: Steuerung UpdateSecurity
	Sicherheitszentrum: UpdateSecurity HubConfiguration

Servicepräfix	Aktionen
securitylake	Sicherheitssee: CreateAws LogSource Sicherheitssee: CreateCustom LogSource securitylake: Abonnement CreateData LakeException securitylake: Konfiguration CreateData LakeOrganization Securitylake: CreateSubscriber securitylake: Benachrichtigung CreateSubscriber securitylake: DeleteAws LogSource Sicherheitssee: DeleteCustom LogSource securitylake: Abonnement DeleteData LakeException securitylake: Konfiguration DeleteData LakeOrganization Securitylake: DeleteSubscriber securitylake: Benachrichtigung DeleteSubscriber securitylake: Administrator DeregisterData LakeDelegated securitylake: Abonnement GetData LakeException securitylake: Konfiguration GetData LakeOrganization Securitylake: GetData LakeSources Sicherheitssee: GetSubscriber securitylake: Seen ListData Securitylake: Quellen ListLog SecurityLake: ListSubscribers securitylake: Administrator RegisterData LakeDelegated

Servicepräfix	Aktionen
	securitylake: Abonnement UpdateData LakeException Securitylake: UpdateSubscriber securitylake: Benachrichtigung UpdateSubscriber
serverlessrepo	serverloses Repo: CreateApplication serverlessrepo: Version CreateApplication serverlessrepo: CreateCloud FormationChange Satz serverlessrepo: CreateCloud FormationTemplate serverloses Repo: DeleteApplication serverloses Repo: GetApplication serverlessrepo: Richtlinie GetApplication serverlessrepo: GetCloud FormationTemplate serverlessrepo: Abhängigkeiten ListApplication serverlessrepo: ListApplications serverlessrepo: Versionen ListApplication serverlessrepo: PutApplication Richtlinie serverlessrepo: UnshareApplication serverloses Repo: UpdateApplication

Servicepräfix	Aktionen
servicecatalog	Servicekatalog: Teilen AcceptPortfolio Servicekatalog: AssociateBudget WithResource Servicekatalog: AssociatePrincipal WithPortfolio Servicekatalog: AssociateProduct WithPortfolio Servicekatalog: AssociateService ActionWith ProvisioningArtifact Servicekatalog: Artifact BatchAssociate ServiceAction WithProvisioning Servicekatalog: Artifact BatchDisassociate ServiceAction FromProvisioning Servicekatalog: CopyProduct Servicekatalog: CreateConstraint Servicekatalog: CreatePortfolio Servicekatalog: Teilen CreatePortfolio Servicekatalog: CreateProduct Servicekatalog: CreateProvisioned ProductPlan Servicekatalog: Artifact CreateProvisioning ServicekatalogCreateService: Aktion Servicekatalog: DeleteConstraint Servicekatalog: DeletePortfolio Servicekatalog: Teilen DeletePortfolio Servicekatalog: DeleteProduct Servicekatalog: DeleteProvisioned ProductPlan

Servicepräfix	Aktionen
	Servicekatalog: Artifact DeleteProvisioning ServicekatalogDeleteService: Aktion Servicekatalog: DescribeConstraint Servicekatalog: DescribeCopy ProductStatus Servicekatalog: DescribePortfolio Servicekatalog: Aktien DescribePortfolio Servicekatalog: DescribePortfolio ShareStatus Servicekatalog: DescribeProduct Servicekatalog: DescribeProduct AsAdmin Servicekatalog: Ansicht DescribeProduct Servicekatalog: DescribeProvisioned ProductPlan Servicekatalog: Artifact DescribeProvisioning ServicekatalogDescribeProvisioning: Parameter Servicekatalog: DescribeRecord Servicekatalog: Aktion DescribeService servicecatalog: Parameter DescribeService ActionExecution Servicekatalog: deaktivieren AWSOrganizationsAccess Servicekatalog: DisassociateBudget FromResource Servicekatalog: DisassociatePrincipal FromPortfolio Servicekatalog: DisassociateProduct FromPortfolio

Servicepräfix	Aktionen
	Servicekatalog: DisassociateService ActionFrom ProvisioningArtifact Servicekatalog: aktivieren AWSOrganizationsAccess Servicekatalog: ExecuteProvisioned ProductPlan Servicekatalog: Aktion ExecuteProvisioned ProductService Servicekatalog: GET AWSOrganizationsAccessStatus Servicekatalog: GetProvisioned ProductOutputs Servicekatalog: ImportAs ProvisionedProduct Servicekatalog: ListAccepted PortfolioShares Servicekatalog: ListBudgets ForResource Servicekatalog: ListConstraints ForPortfolio servicecatalog: Pfade ListLaunch Servicekatalog: ListOrganization PortfolioAccess Servicekatalog: Zugriff ListPortfolio Servicekatalog: ListPortfolios Servicekatalog: ListPortfolios ForProduct Servicekatalog: ListPrincipals ForPortfolio Servicekatalog: ListProvisioned ProductPlans Servicekatalog: Artefakte ListProvisioning Servicekatalog: ListProvisioning ArtifactsFor ServiceAction Servicekatalog: Geschichte ListRecord

Servicepräfix	Aktionen
	Servicekatalog: Aktionen ListService Servicekatalog: ListService ActionsFor ProvisioningArtifact Servicekatalog: ListStack InstancesFor ProvisionedProduct Servicekatalog: NotifyProvision ProductEngine WorkflowResult servicecatalog: Ergebnis NotifyTerminate ProvisionedProduct EngineWorkflow servicecatalog: Ergebnis NotifyUpdate ProvisionedProduct EngineWorkflow Servicekatalog: ProvisionProduct Servicekatalog: Teilen RejectPortfolio Servicekatalog: Produkte ScanProvisioned Servicekatalog: SearchProducts Servicekatalog: SearchProducts AsAdmin Servicekatalog: Produkte SearchProvisioned Servicekatalog: Produkt TerminateProvisioned Servicekatalog: UpdateConstraint Servicekatalog: UpdatePortfolio Servicekatalog: Teilen UpdatePortfolio Servicekatalog: UpdateProduct Servicekatalog: Produkt UpdateProvisioned Servicekatalog: UpdateProvisioned ProductProperties Servicekatalog: Artifact UpdateProvisioning

Servicepräfix	Aktionen
	ServicekatalogUpdateService: Aktion

Servicepräfix	Aktionen
servicediscovery	servicediscovery: Namespace CreateHttp
	Servicediscovery: CreatePrivate DnsNamespace
	Servicediscovery: CreatePublic DnsNamespace
	Servicediscovery: CreateService
	Servicediscovery: DeleteNamespace
	Servicediscovery: DeleteService
	Servicediscovery: DeregisterInstance
	Servicediscovery: GetInstance
	Servicediscovery: GetInstances HealthStatus
	Servicediscovery: GetNamespace
	Servicediscovery: GetOperation
	Servicediscovery: GetService
	Servicediscovery: ListInstances
	Servicediscovery: ListNamespaces
	Servicediscovery: ListOperations
	Servicediscovery: ListServices
	Servicediscovery: RegisterInstance
	servicediscovery: Namespace UpdateHttp
	servicediscovery: Status UpdateInstance CustomHealth
	Servicediscovery: UpdatePrivate DnsNamespace
	Servicediscovery: UpdatePublic DnsNamespace

Servicepräfix	Aktionen
	Servicediscovery: UpdateService
servicequotas	Serviceangebote: AssociateService QuotaTemplate servicequotas: Vorlage DeleteService QuotaIncrease RequestFrom Servicequoten: DisassociateService QuotaTemplate Serviceangebote: GetAssociation ForService QuotaTemplate Serviceangebote: GET AWSDefaultServiceQuota Serviceangebote: Ändern GetRequested ServiceQuota servicequotas: Kontingent GetService servicequotas: Vorlage GetService QuotaIncrease RequestFrom Servicequoten: Liste AWSDefaultServiceQuotas Serviceangebote: ListRequested ServiceQuota ChangeHistory Serviceangebote: ListRequested ServiceQuota ChangeHistory ByQuota servicequotas: Vorlage ListService QuotaIncrease RequestsIn servicequotas: Kontingente ListService Servicequoten: ListServices servicequotas: Vorlage PutService QuotaIncrease RequestInto Servicequoten: RequestService QuotaIncrease

Servicepräfix	Aktionen
ses	verwendet: BatchGet MetricData sieht: CloneReceipt RuleSet ses: CreateConfiguration Satz ses: CreateConfiguration SetEvent Ziel ses: CreateConfiguration SetTracking Optionen siehe: CreateContact ses: CreateContact Liste ses: CreateCustom VerificationEmail Vorlage ses: CreateDedicated IpPool sieht: CreateDeliverability TestReport ses: CreateEmail Identität siehe: CreateEmail IdentityPolicy ses: CreateEmail Vorlage ses: CreateImport Job ses: CreateReceipt Filter ses: CreateReceipt Regel siehe: CreateReceipt RuleSet sieht: CreateTemplate ses: DeleteConfiguration Satz ses: DeleteConfiguration SetEvent Ziel ses: DeleteConfiguration SetTracking Optionen

Servicepräfix	Aktionen
	siehe: DeleteContact
	ses: DeleteContact Liste
	ses: DeleteCustom VerificationEmail Vorlage
	ses: DeleteDedicated IpPool
	ses: DeleteEmail Identität
	siehe: DeleteEmail IdentityPolicy
	ses: DeleteEmail Vorlage
	ses: DeleteIdentity
	ses: DeleteIdentity Politik
	ses: DeleteReceipt Filter
	ses: DeleteReceipt Regel
	siehe: DeleteReceipt RuleSet
	ses: DeleteSuppressed Reiseziel
	siehe: DeleteTemplate
	sieht: DeleteVerified EmailAddress
	ses: DescribeActive ReceiptRule Satz
	ses: DescribeConfiguration Satz
	ses: DescribeReceipt Regel
	siehe: DescribeReceipt RuleSet
	sieht: GetAccount
	sieht: GetAccount SendingEnabled

Servicepräfix	Aktionen
	ses: GetBlacklist Berichte
	ses: GetConfiguration Set
	ses: GetConfiguration SetEvent Ziele
	siehe: GetContact
	ses: GetContact Liste
	ses: GetCustom VerificationEmail Vorlage
	ses: GetDedicated Up
	siehe: GetDedicated IpPool
	siehe: GetDedicated Ips
	siehe: GetDeliverability DashboardOptions
	sieht: GetDeliverability TestReport
	sieht: GetDomain DeliverabilityCampaign
	sieht: GetDomain StatisticsReport
	ses: GetEmail Identität
	siehe: GetEmail IdentityPolicies
	ses: GetEmail Vorlage
	ses: GetIdentity DkimAttributes
	sieht: GetIdentity MailFrom DomainAttributes
	sieht: GetIdentity NotificationAttributes
	ses: GetIdentity Richtlinien
	siehe: GetIdentity VerificationAttributes

Servicepräfix	Aktionen
	ses: GetImport Job
	ses: GetMessage Einblicke
	ses: GetSend Kontingent
	ses: GetSend Statistik
	ses: GetSuppressed Reiseziel
	siehe: GetTemplate
	ses: ListConfiguration Sets
	ses: ListContact Listen
	siehe: ListContacts
	ses: ListCustom VerificationEmail Vorlagen
	siehe: ListDedicated IpPools
	sieht: ListDeliverability TestReports
	sieht: ListDomain DeliverabilityCampaigns
	ses: ListEmail Identitäten
	ses: Vorlagen ListEmail
	ses: ListExport Jobs
	ses: ListIdentities
	ses: ListIdentity Richtlinien
	ses: ListImport Jobs
	ses: ListReceipt Filter
	verwendet: ListReceipt RuleSets

Servicepräfix	Aktionen
	sieht: ListRecommendations
	ses: ListSuppressed Destinationen
	siehe: ListTemplates
	sieht: ListVerified EmailAddresses
	sieht: PutAccount DedicatedIp WarmupAttributes
	ses: PutAccount Einzelheiten
	siehe: PutAccount SendingAttributes
	sieht: PutAccount SuppressionAttributes
	sieht: PutAccount VdmAttributes
	ses: PutConfiguration SetDelivery Optionen
	ses: PutConfiguration SetReputation Optionen
	ses: PutConfiguration SetSending Optionen
	ses: PutConfiguration SetSuppression Optionen
	ses: PutConfiguration SetTracking Optionen
	ses: PutConfiguration SetVdm Optionen
	ses: PutDedicated IpIn Schwimmbad
	Meer: PutDedicated IpPool ScalingAttributes
	ses: PutDedicated IpWarmup Attribute
	siehe: PutDeliverability DashboardOption
	sieht: PutEmail IdentityConfiguration SetAttributes
	ses: PutEmail IdentityDkim Attribute

Servicepräfix	Aktionen
	siehe: PutEmail IdentityDkim SigningAttributes
	ses: PutEmail IdentityFeedback Attribute
	siehe: PutEmail IdentityMail FromAttributes
	ses: PutIdentity Politik
	ses: PutSuppressed Reiseziel
	siehe: ReorderReceipt RuleSet
	sieht: SendBounce
	sieht: SendCustom VerificationEmail
	ses: SetActive ReceiptRule Satz
	siehe: SetIdentity DkimEnabled
	ses: SetIdentity FeedbackForwarding Aktiviert
	ses: SetIdentity HeadersIn NotificationsEnabled
	ses: SetIdentity MailFrom Domäne
	ses: SetIdentity NotificationTopic
	sieht: SetReceipt RulePosition
	sieht: TestRender EmailTemplate
	ses: TestRender Vorlage
	ses: UpdateAccount SendingEnabled
	ses: UpdateConfiguration SetEvent Reiseziel
	siehe: UpdateConfiguration SetReputation MetricsEnabled
	ses: UpdateConfiguration SetSending Aktiviert

Servicepräfix	Aktionen
	ses: UpdateConfiguration SetTracking Optionen siehe: UpdateContact ses: UpdateContact Liste ses: UpdateCustom VerificationEmail Vorlage ses: UpdateEmail IdentityPolicy ses: UpdateEmail Vorlage ses: UpdateReceipt Regel siehe: UpdateTemplate siehe: VerifyDomain Diim ses: Identität VerifyDomain ses: VerifyEmail Adresse ses: VerifyEmail Identität

Servicepräfix	Aktionen
shield	Schild: Assoziierte RT LogBucket shield: Überprüfen AssociateHealth Schild: AssociateProactive EngagementDetails Schild: CreateProtection Schild: CreateProtection Gruppe Schild: CreateSubscription Schild: DeleteProtection Schild: DeleteProtection Gruppe Schild: DeleteSubscription Schild: DescribeAttack Schild: DescribeAttack Statistiken shield:DescribeDRTAccess Schild: DescribeEmergency ContactSettings Schild: DescribeProtection Schild: DescribeProtection Gruppe Schild: DescribeSubscription Schild: DisableApplication LayerAutomatic Antwort Schild: DisableProactive Engagement Schild: Nicht assoziiertes RT LogBucket shield:DisassociateDRTRole shield: Überprüfen DisassociateHealth

Servicepräfix	Aktionen
	Schild: EnableApplication LayerAutomatic Antwort Schild: EnableProactive Engagement Schild: GetSubscription Staat Schild: ListAttacks Schild: ListProtection Gruppen Schild: ListProtections Schild: ListResources InProtection Gruppe Schild: UpdateApplication LayerAutomatic Antwort Schild: UpdateEmergency ContactSettings Schild: UpdateProtection Gruppe Schild: UpdateSubscription

Servicepräfix	Aktionen
signer	Unterzeichner: Genehmigung AddProfile
	Unterzeichner: Profil CancelSigning
	Unterzeichner: Job DescribeSigning
	Unterzeichner: Status GetRevocation
	Unterzeichner: Plattform GetSigning
	Unterzeichner: Profil GetSigning
	Unterzeichner: Berechtigungen ListProfile
	Unterzeichner: Jobs ListSigning
	Unterzeichner: Plattformen ListSigning
	Unterzeichner: Profile ListSigning
	Unterzeichner: Profil PutSigning
	Unterzeichner: Erlaubnis RemoveProfile
	Unterzeichner: RevokeSignature
	Unterzeichner: Profil RevokeSigning
	Unterzeichner: SignPayload
	Unterzeichner: Job StartSigning

Servicepräfix	Aktionen
simspaceweaver	Simspaceweaver: CreateSnapshot Simspaceweaver: DeleteApp Simspaceweaver: DeleteSimulation Simspaceweaver: DescribeApp Simspaceweaver: DescribeSimulation Simspaceweaver: ListApps Simspaceweaver: ListSimulations Simspaceweaver: StartApp Simspaceweaver: StartClock Simspaceweaver: StartSimulation Simspaceweaver: StopApp Simspaceweaver: StopClock Simspaceweaver: StopSimulation

Servicepräfix	Aktionen
sms	sms: CreateApp
	sms: CreateReplication Job
	sms: DeleteApp
	sms: DeleteApp LaunchConfiguration
	sms: DeleteApp ReplicationConfiguration
	sms: DeleteApp ValidationConfiguration
	sms: DeleteReplication Job
	sms: DeleteServer Katalog
	sms: DisassociateConnector
	sms: GenerateChange Eingestellt
	sms: GenerateTemplate
	sms: GetApp
	sms: GetApp LaunchConfiguration
	sms: GetApp ReplicationConfiguration
	sms: GetApp ValidationConfiguration
	sms: GetApp ValidationOutput
	sms: GetConnectors
	sms: GetReplication Jobs
	sms: GetReplication Läuft
	sms: GetServers
	sms: ImportApp Katalog

Servicepräfix	Aktionen
	sms: ImportServer Katalog
	sms: LaunchApp
	sms: ListApps
	sms: NotifyApp ValidationOutput
	sms: PutApp LaunchConfiguration
	sms: PutApp ReplicationConfiguration
	sms: PutApp ValidationConfiguration
	sms: StartApp Replikation
	sms: StartOn DemandApp Replikation
	sms: StartOn DemandReplication Ausführen
	sms: StopApp Replikation
	sms: TerminateApp
	sms: UpdateApp
	sms: UpdateReplication Job

Servicepräfix	Aktionen
sms-voice	sms-voice: Konfiguration AssociateProtect sms-voice: Eingestellt CreateConfiguration sms-voice: Ziel CreateConfiguration SetEvent sms-voice: Ziel CreateEvent SMS-Sprache: CreateOpt OutList SMS-Stimme: CreatePool sms-voice: Konfiguration CreateProtect SMS-Stimme: CreateRegistration sms-voice: Verband CreateRegistration sms-voice: Anlage CreateRegistration sms-voice: Version CreateRegistration SMS-Stimme: CreateVerified DestinationNumber sms-voice: Konfiguration DeleteAccount DefaultProtect sms-voice: Eingestellt DeleteConfiguration sms-voice: Ziel DeleteConfiguration SetEvent SMS-Sprache: DeleteDefault MessageType SMS-Stimme: DeleteDefault SenderId sms-voice: Ziel DeleteEvent SMS-Sprache: DeleteKeyword SMS-Stimme: DeleteMedia MessageSpend LimitOverride SMS-Stimme: DeleteOpted OutNumber

Servicepräfix	Aktionen
	SMS-Stimme: DeleteOpt OutList SMS-Stimme: DeletePool sms-voice: Konfiguration DeleteProtect SMS-Stimme: DeleteRegistration sms-voice: Anlage DeleteRegistration SMS-Stimme: DeleteText MessageSpend LimitOverride SMS-Stimme: DeleteVerified DestinationNumber SMS-Stimme: DeleteVoice MessageSpend LimitOverride sms-voice: Attribute DescribeAccount sms-voice: Grenzwerte DescribeAccount sms-voice: Legt fest DescribeConfiguration SMS-Sprache: DescribeKeywords SMS-Stimme: DescribeOpted OutNumbers SMS-Stimme: DescribeOpt OutLists sms-voice: Zahlen DescribePhone SMS-Stimme: DescribePools sms-voice: Konfigurationen DescribeProtect sms-voice: Anlagen DescribeRegistration SMS-Voice: DescribeRegistration FieldDefinitions SMS-Stimme: DescribeRegistration FieldValues SMS-Stimme: DescribeRegistrations

Servicepräfix	Aktionen
	SMS-Stimme: DescribeRegistration SectionDefinitions SMS-Stimme: DescribeRegistration TypeDefinitions sms-voice: Versionen DescribeRegistration sms-voice: IDs DescribeSender sms-voice: Grenzwerte DescribeSpend SMS-Sprache: DescribeVerified DestinationNumbers sms-voice: Identität DisassociateOrigination sms-voice: Konfiguration DisassociateProtect sms-voice: Version DiscardRegistration sms-voice: Ziele GetConfiguration SetEvent SMS-Sprache: GetProtect ConfigurationCountry RuleSet sms-voice: Sätze ListConfiguration SMS-Sprache: ListPool OriginationIdentities sms-voice: Verbände ListRegistration sms-voice: PutKeyword SMS-Stimme: PutOpted OutNumber sms-voice: Nummer ReleasePhone sms-voice: ID ReleaseSender sms-voice: Nummer RequestPhone sms-voice: ID RequestSender sms-voice: Code SendDestination NumberVerification

Servicepräfix	Aktionen
	sms-voice: Konfiguration SetAccount DefaultProtect SMS-Stimme: SetDefault MessageType SMS-Stimme: SetDefault SenderId SMS-Stimme: SetMedia MessageSpend LimitOverride SMS-Stimme: SetText MessageSpend LimitOverride SMS-Stimme: SetVoice MessageSpend LimitOverride SMS-Stimme: Version SubmitRegistration sms-voice: Ziel UpdateConfiguration SetEvent sms-voice: Ziel UpdateEvent sms-voice: Nummer UpdatePhone sms-voice: UpdatePool sms-voice: Konfiguration UpdateProtect SMS-Stimme: UpdateProtect ConfigurationCountry RuleSet SMS-Stimme: ID UpdateSender

Servicepräfix	Aktionen
snowball	Schneeball: CancelCluster
	Schneeball: CancelJob
	Schneeball: CreateAddress
	Schneeball: CreateCluster
	Schneeball: CreateJob
	Schneeball: CreateLong TermPricing
	Schneeball: CreateReturn ShippingLabel
	Schneeball: DescribeAddress
	Schneeball: DescribeAddresses
	Schneeball: DescribeCluster
	Schneeball: DescribeJob
	Schneeball: DescribeReturn ShippingLabel
	Schneeball: Manifest GetJob
	Schneeball: GetJob UnlockCode
	Schneeball: Verwendung GetSnowball
	Schneeball: Aktualisierungen GetSoftware
	Schneeball: Jobs ListCluster
	Schneeball: ListClusters
	Schneeball: Bilder ListCompatible
	Schneeball: ListJobs
	Schneeball: ListLong TermPricing

Servicepräfix	Aktionen
	Schneeball: Standorte ListPickup Schneeball: Versionen ListService Schneeball: UpdateCluster Schneeball: UpdateJob Schneeball: UpdateJob ShipmentState Schneeball: UpdateLong TermPricing
sqs	sägt: AddPermission sqs: CancelMessage MoveTask sqs: CreateQueue sqs: DeleteQueue sqs: PurgeQueue sqs: RemovePermission sqs: Attribute SetQueue

Servicepräfix	Aktionen
ssm	ssm: Artikel AssociateOps ItemRelated
	ssm: CancelCommand
	ssm: CancelMaintenance WindowExecution
	ssm: CreateActivation
	ssm: CreateAssociation
	ssm: Batch CreateAssociation
	ssm: CreateDocument
	ssm: Fenster CreateMaintenance
	ssm: Artikel CreateOps
	ssm: Metadaten CreateOps
	ssm: Basislinie CreatePatch
	ssm: CreateResource DataSync
	ssm: DeleteActivation
	ssm: DeleteAssociation
	ssm: DeleteDocument
	ssm: DeleteInventory
	ssm: Fenster DeleteMaintenance
	ssm: Artikel DeleteOps
	ssm: Metadaten DeleteOps
	ssm: DeleteParameter
	ssm: DeleteParameters

Servicepräfix	Aktionen
	ssm: Ausgangswert DeletePatch
	ssm: DeleteResource DataSync
	ssm: Richtlinie DeleteResource
	ssm: Instanz DeregisterManaged
	ssm: DeregisterPatch BaselineFor PatchGroup
	ssm: Fenster DeregisterTarget FromMaintenance
	ssm: Fenster DeregisterTask FromMaintenance
	ssm: DescribeActivations
	ssm: DescribeAssociation
	ssm: Hinrichtungen DescribeAssociation
	ssm: DescribeAssociation ExecutionTargets
	ssm: Hinrichtungen DescribeAutomation
	ssm: DescribeAutomation StepExecutions
	ssm: Patches DescribeAvailable
	ssm: DescribeDocument
	ssm: Parameter DescribeDocument
	ssm: Erlaubnis DescribeDocument
	ssm: DescribeEffective InstanceAssociations
	ssm: DescribeEffective PatchesFor PatchBaseline
	ssm: DescribeInstance AssociationsStatus
	ssm: Informationen DescribeInstance

Servicepräfix	Aktionen
	ssm: Patches DescribeInstance
	ssm: DescribeInstance PatchStates
	ssm: Gruppe DescribeInstance PatchStates ForPatch
	ssm: Eigenschaften DescribeInstance
	ssm: Löschungen DescribeInventory
	ssm: DescribeMaintenance WindowExecutions
	ssm: DescribeMaintenance WindowExecution TaskInvocations
	ssm: Aufgaben DescribeMaintenance WindowExecution
	ssm: Windows DescribeMaintenance
	ssm: DescribeMaintenance WindowSchedule
	ssm: Ziel DescribeMaintenance WindowsFor
	ssm: DescribeMaintenance WindowTargets
	ssm: DescribeMaintenance WindowTasks
	ssm: Artikel DescribeOps
	ssm: DescribeParameters
	ssm: Grundlinien DescribePatch
	ssm: Gruppen DescribePatch
	ssm: DescribePatch GroupState
	ssm: Eigenschaften DescribePatch
	ssm: DescribeSessions
	ssm: Artikel DisassociateOps ItemRelated

Servicepräfix	Aktionen
	ssm: Ausführung GetAutomation
	ssm: Bundesstaat GetCalendar
	ssm: Aufruf GetCommand
	ssm: Status GetConnection
	ssm: GetDefault PatchBaseline
	ssm: GetDeployable PatchSnapshot ForInstance
	ssm: GetDocument
	ssm: GetInventory
	ssm: Schema GetInventory
	ssm: Fenster GetMaintenance
	ssm: GetMaintenance WindowExecution
	ssm: Aufgabe GetMaintenance WindowExecution
	ssm: GetMaintenance WindowExecution TaskInvocation
	ssm: GetMaintenance WindowTask
	ssm: Artikel GetOps
	ssm: Metadaten GetOps
	ssm: Zusammenfassung GetOps
	ssm: GetParameter
	ssm: Geschichte GetParameter
	ssm: GetParameters
	ssm: GetParameters ByPath

Servicepräfix	Aktionen
	ssm: Ausgangswert GetPatch
	ssm: GetPatch BaselineFor PatchGroup
	ssm: Richtlinien GetResource
	ssm: Einstellung GetService
	ssm: Version LabelParameter
	ssm: ListAssociations
	ssm: Versionen ListAssociation
	ssm: Aufrufe ListCommand
	ssm: ListCommands
	ssm: Artikel ListCompliance
	ssm: Zusammenfassungen ListCompliance
	ssm: ListDocument MetadataHistory
	ssm: ListDocuments
	ssm: Versionen ListDocument
	ssm: Verbände ListInstance
	ssm: Einträge ListInventory
	ssm: ListOps ItemEvents
	ssm: Artikel ListOps ItemRelated
	ssm: Metadaten ListOps
	ssm: ListResource ComplianceSummaries
	ssm: ListResource DataSync

Servicepräfix	Aktionen
	ssm: Erlaubnis ModifyDocument
	ssm: Artikel PutCompliance
	ssm: PutInventory
	ssm: PutParameter
	ssm: Richtlinie PutResource
	ssm: RegisterDefault PatchBaseline
	ssm: Instanz RegisterManaged
	ssm: RegisterPatch BaselineFor PatchGroup
	ssm: Fenster RegisterTarget WithMaintenance
	ssm: Fenster RegisterTask WithMaintenance
	ssm: Einstellung ResetService
	ssm: ResumeSession
	ssm: Signal SendAutomation
	ssm: SendCommand
	ssm: Einmal StartAssociations
	ssm: Ausführung StartAutomation
	ssm: StartChange RequestExecution
	ssm: StartSession
	ssm: Ausführung StopAutomation
	ssm: TerminateSession
	ssm: Ausführung UnlabelParameter

Servicepräfix	Aktionen
	ssm: UpdateAssociation
	ssm: Status UpdateAssociation
	ssm: UpdateDocument
	ssm: UpdateDocument DefaultVersion
	ssm: Metadaten UpdateDocument
	ssm: Informationen UpdateInstance
	ssm: Fenster UpdateMaintenance
	ssm: UpdateMaintenance WindowTarget
	ssm: UpdateMaintenance WindowTask
	ssm: UpdateManaged InstanceRole
	ssm: Artikel UpdateOps
	ssm: Metadaten UpdateOps
	ssm: Basislinie UpdatePatch
	ssm: UpdateResource DataSync
	ssm: Einstellung UpdateService

Servicepräfix	Aktionen
ssm-incidents	SSM-Vorfälle: BatchGet IncidentFindings ssm-incidents: Set CreateReplication ssm-incidents: Plan CreateResponse ssm-incidents: Ereignis CreateTimeline ssm-incidents: Rekord DeleteIncident ssm-incidents: Set DeleteReplication ssm-incidents: Richtlinie DeleteResource SSM-Vorfälle: Plan DeleteResponse ssm-incidents: Ereignis DeleteTimeline ssm-incidents: Rekord GetIncident ssm-incidents: Set GetReplication ssm-incidents: Richtlinien GetResource SSM-Vorfälle: Plan GetResponse ssm-incidents: Ereignis GetTimeline SSM-Vorfälle: Ergebnisse ListIncident SSM-Vorfälle: Aufzeichnungen ListIncident SSM-Vorfälle: Artikel ListRelated ssm-incidents: Sätze ListReplication ssm-incidents: Pläne ListResponse ssm-incidents: Ereignisse ListTimeline SSM-Vorfälle: Richtlinie PutResource

Servicepräfix	Aktionen
	SSM-Vorfälle: StartIncident SSM-Vorfälle: Schutz UpdateDeletion SSM-Vorfälle: Rekord UpdateIncident SSM-Vorfälle: Artikel UpdateRelated ssm-incidents: Satz UpdateReplication ssm-incidents: Plan UpdateResponse ssm-incidents: Ereignis UpdateTimeline

Servicepräfix	Aktionen
ssm-sap	ssm-sap: BackupDatabase
	ssm-sap: Erlaubnis DeleteResource
	ssm-sap: DeregisterApplication
	ssm-sap: GetApplication
	ssm-sap: GetComponent
	ssm-sap: GetDatabase
	ssm-sap: GetOperation
	ssm-sap: Erlaubnis GetResource
	ssm-sap: ListApplications
	ssm-sap: ListComponents
	ssm-sap: ListDatabases
	ssm-sap: Veranstaltungen ListOperation
	ssm-sap: ListOperations
	ssm-sap: Erlaubnis PutResource
	ssm-sap: RegisterApplication
	ssm-sap: RestoreDatabase
	ssm-sap: StartApplication
	ssm-sap: Aktualisieren StartApplication
	ssm-sap: StopApplication
	ssm-sap: Einstellungen UpdateApplication
	ssm-Sap: Hana aktualisieren BackupSettings

Servicepräfix	Aktionen
Status	erklärt: CreateActivity Staaten: CreateState Maschine Staaten: CreateState MachineAlias Staaten: DeleteActivity Staaten: DeleteState Maschine Staaten: DeleteState MachineAlias Staaten: DeleteState MachineVersion Staaten: DescribeActivity Staaten: DescribeExecution Staaten: DescribeMap Lauf Staaten: DescribeState Maschine Staaten: DescribeState MachineAlias Staaten: DescribeState MachineFor Hinrichtung Staaten: GetExecution Geschichte Staaten: ListActivities Staaten: ListExecutions Staaten: ListMap Läuft Staaten: ListState MachineAliases Staaten: ListState Maschinen Staaten: ListState MachineVersions Staaten: SendTask Fehlschlag

Servicepräfix	Aktionen
	Staaten: SendTask Heartbeat Staaten: Erfolg SendTask Staaten: StartExecution Staaten: StopExecution Staaten: UpdateMap Lauf Staaten: UpdateState Maschine Staaten: UpdateState MachineAlias Staaten: ValidateState MachineDefinition
sts	sts: AssumeRole sts: AssumeRole mit SAML sts: Identität AssumeRole WithWeb sts: DecodeAuthorization Nachricht sts: GetAccess KeyInfo sts: GetCaller Identität sts: GetFederation Token sts: GetSession Wertmarke

Servicepräfix	Aktionen
swf	swf: DeprecateActivity Typ swf: DeprecateDomain swf: DeprecateWorkflow Typ swf: DescribeActivity Typ swf: DescribeDomain swf: DescribeWorkflow Typ swf: ListActivity Typen swf: ListDomains swf: ListWorkflow Typen swf: RegisterActivity Typ swf: RegisterDomain swf: RegisterWorkflow Typ swf: UndeprecateActivity Typ swf: UndeprecateDomain swf: UndeprecateWorkflow Typ

Servicepräfix	Aktionen
synthetics	Kunststoffe: AssociateResource
	Kunststoffe: CreateCanary
	Kunststoffe: CreateGroup
	Kunststoffe: DeleteCanary
	Kunststoffe: DeleteGroup
	Kunststoffe: DescribeCanaries
	Kunststoffe: DescribeCanaries LastRun
	Kunststoffe: Versionen DescribeRuntime
	Kunststoffe: DisassociateResource
	Kunststoffe: GetCanary
	Kunststoffe: Läuft GetCanary
	Kunststoffe: GetGroup
	Kunststoffe: Gruppen ListAssociated
	Kunststoffe: Ressourcen ListGroup
	Kunststoffe: ListGroups
	Kunststoffe: StartCanary
	Kunststoffe: StopCanary
	Kunststoffe: UpdateCanary

Servicepräfix	Aktionen
Tag (Markierung)	Etikett: Schöpfung DescribeReport Etikett: GetCompliance Zusammenfassung Etikett: GetResources Etikett: StartReport Schöpfung

Servicepräfix	Aktionen
textract/	textrahieren: AnalyzeDocument
	textrahieren: AnalyzeExpense
	textract:AnalyzeID
	textrahieren: CreateAdapter
	textract: Version CreateAdapter
	textrahieren: DeleteAdapter
	textract: Version DeleteAdapter
	textract: Text DetectDocument
	textrahieren: GetAdapter
	textract: Version GetAdapter
	textract: Analyse GetDocument
	textrahieren: GetDocument TextDetection
	textract: Analyse GetExpense
	textract: Analyse GetLending
	textrahieren: GetLending AnalysisSummary
	textrahieren: ListAdapters
	textract: Versionen ListAdapter
	textract: Analyse StartDocument
	textrahieren: StartDocument TextDetection
	textract: Analyse StartExpense
	textract: Analyse StartLending

Servicepräfix	Aktionen
	textrahieren: UpdateAdapter

Servicepräfix	Aktionen
timestream	Zeitstrom: CancelQuery Zeitstrom: CreateDatabase timestream: Abfrage CreateScheduled Zeitstrom: CreateTable Zeitstrom: DeleteDatabase timestream: Abfrage DeleteScheduled Zeitstrom: DeleteTable timestream: Einstellungen DescribeAccount Timestream: DescribeDatabase timestream: Abfrage DescribeScheduled Zeitstrom: DescribeTable timestream: Abfrage ExecuteScheduled Zeitstrom: ListBatch LoadTasks Zeitstrom: ListDatabases timestream: Abfragen ListScheduled Zeitstrom: ListTables Zeitstrom: PrepareQuery timestream: Einstellungen UpdateAccount Timestream: UpdateDatabase timestream: Abfrage UpdateScheduled Zeitstrom: UpdateTable

Servicepräfix	Aktionen
tnb	tnb: CancelSol NetworkOperation
	tnb: CreateSol FunctionPackage
	tnb: CreateSol NetworkInstance
	tnb: CreateSol NetworkPackage
	tnb: DeleteSol FunctionPackage
	tnb: DeleteSol NetworkInstance
	tnb: DeleteSol NetworkPackage
	tnb: GetSol FunctionInstance
	tnb: GetSol FunctionPackage
	tnb: Inhalt GetSol FunctionPackage
	tnb: Deskriptor GetSol FunctionPackage
	tnb: GetSol NetworkInstance
	tnb: GetSol NetworkOperation
	tnb: GetSol NetworkPackage
	tnb: Inhalt GetSol NetworkPackage
	tnb: Deskriptor GetSol NetworkPackage
	tnb: InstantiateSol NetworkInstance
	tnb: ListSol FunctionInstances
	tnb: ListSol FunctionPackages
	tnb: ListSol NetworkInstances
	tnb: ListSol NetworkOperations

Servicepräfix	Aktionen
	tnb: ListSol NetworkPackages
	tnb: Inhalt PutSol FunctionPackage
	tnb: Inhalt PutSol NetworkPackage
	tnb: TerminateSol NetworkInstance
	tnb: UpdateSol FunctionPackage
	tnb: UpdateSol NetworkInstance
	tnb: UpdateSol NetworkPackage
	tnb: Inhalt ValidateSol FunctionPackage
	tnb: Inhalt ValidateSol NetworkPackage

Servicepräfix	Aktionen
transcribe	transkribieren: CreateCall AnalyticsCategory transkribieren: Modell CreateLanguage transkribieren: Wortschatz CreateMedical transkribieren: CreateVocabulary transkribieren: Filter CreateVocabulary transkribieren: DeleteCall AnalyticsCategory transkribieren: DeleteCall AnalyticsJob transkribieren: Modell DeleteLanguage transkribieren: DeleteMedical ScribeJob transkribieren: DeleteMedical TranscriptionJob transkribieren: Wortschatz DeleteMedical transkribieren: Job DeleteTranscription transkribieren: DeleteVocabulary transkribieren: Filter DeleteVocabulary transkribieren: Modell DescribeLanguage transkribieren: GetCall AnalyticsCategory transkribieren: GetCall AnalyticsJob transkribieren: GetMedical ScribeJob transkribieren: GetMedical TranscriptionJob transkribieren: Wortschatz GetMedical transkribieren: Job GetTranscription

Servicepräfix	Aktionen
	transkribieren: GetVocabulary transkribieren: Filter GetVocabulary transkribieren: ListCall AnalyticsCategories transkribieren: ListCall AnalyticsJobs transkribieren: Modelle ListLanguage transkribieren: ListMedical ScribeJobs transkribieren: ListMedical TranscriptionJobs transkribieren: Vokabeln ListMedical transkribieren: Jobs ListTranscription transkribieren: ListVocabularies transkribieren: Filter ListVocabulary transkribieren: StartCall AnalyticsJob transkribieren: Transkription StartCall AnalyticsStream transkribieren: Socket StartCall AnalyticsStream TranscriptionWeb transkribieren: StartMedical ScribeJob transkribieren: StartMedical StreamTranscription transkribieren: StartMedical StreamTranscription WebSocket transkribieren: StartMedical TranscriptionJob transkribieren: Transkription StartStream transkribieren: Socket StartStream TranscriptionWeb transkribieren: Job StartTranscription

Servicepräfix	Aktionen
	transkribieren: UpdateCall AnalyticsCategory transkribieren: Wortschatz UpdateMedical transkribieren: UpdateVocabulary transkribieren: Filter UpdateVocabulary

Servicepräfix	Aktionen
Übertragung	übertragen: CreateAccess
	Übertragung: CreateAgreement
	Übertragung: CreateConnector
	Übertragung: CreateProfile
	Übertragung: CreateServer
	Übertragung: CreateUser
	Übertragung: CreateWorkflow
	Übertragung: DeleteAccess
	Übertragung: DeleteAgreement
	Übertragung: DeleteCertificate
	Übertragung: DeleteConnector
	Übertragung: DeleteHost Schlüssel
	Übertragung: DeleteProfile
	Übertragung: DeleteServer
	Übertragung: DeleteSsh PublicKey
	Übertragung: DeleteUser
	Übertragung: DeleteWorkflow
	Übertragung: DescribeAccess
	Übertragung: DescribeAgreement
	Übertragung: DescribeCertificate
	Übertragung: DescribeConnector

Servicepräfix	Aktionen
	Übertragung: DescribeExecution
	Übertragung: DescribeHost Schlüssel
	Übertragung: DescribeProfile
	Übertragung: DescribeSecurity Richtlinie
	Übertragung: DescribeServer
	Übertragung: DescribeUser
	Übertragung: DescribeWorkflow
	Übertragung: ImportCertificate
	Übertragung: ImportHost Schlüssel
	Übertragung: ImportSsh PublicKey
	Übertragung: ListAccesses
	Übertragung: ListCertificates
	Übertragung: ListConnectors
	Übertragung: ListExecutions
	Übertragung: ListHost Schlüssel
	Übertragung: ListProfiles
	Übertragung: ListSecurity Richtlinien
	Übertragung: ListServers
	Übertragung: ListUsers
	Übertragung: ListWorkflows
	Übertragung: SendWorkflow StepState

Servicepräfix	Aktionen
	Übertragung: StartDirectory Auflistung
	Übertragung: StartFile Übertragung
	Übertragung: StartServer
	Übertragung: StopServer
	Übertragung: TestConnection
	Übertragung: TestIdentity Anbieter
	Übertragung: UpdateAccess
	Übertragung: UpdateAgreement
	Übertragung: UpdateCertificate
	Übertragung: UpdateConnector
	Übertragung: UpdateHost Schlüssel
	Übertragung: UpdateProfile
	Übertragung: UpdateServer
	Übertragung: UpdateUser

Servicepräfix	Aktionen
translate	übersetze: CreateParallel Daten übersetze: DeleteParallel Daten übersetzen: DeleteTerminology übersetzen: DescribeText TranslationJob übersetze: GetParallel Daten übersetzen: GetTerminology übersetzen: ImportTerminology übersetzen: ListLanguages übersetze: ListParallel Daten übersetzen: ListTerminologies übersetzen: ListText TranslationJobs übersetzen: StartText TranslationJob übersetzen: StopText TranslationJob übersetzen: TranslateDocument übersetzen: TranslateText übersetze: UpdateParallel Daten

Servicepräfix	Aktionen
voiceid	Stimmen-ID: AssociateFraudster
	Stimmen-ID: CreateDomain
	Stimmen-ID: CreateWatchlist
	Stimmen-ID: DeleteDomain
	Stimmen-ID: DeleteFraudster
	Stimmen-ID: DeleteSpeaker
	Stimmen-ID: DeleteWatchlist
	Stimmen-ID: DescribeDomain
	Stimmen-ID: DescribeFraudster
	Stimmen-ID: DescribeFraudster RegistrationJob
	Stimmen-ID: DescribeSpeaker
	Stimmen-ID: DescribeSpeaker EnrollmentJob
	Stimmen-ID: DescribeWatchlist
	Stimmen-ID: DisassociateFraudster
	Stimmen-ID: EvaluateSession
	Stimmen-ID: ListDomains
	Stimmen-ID: ListFraudster RegistrationJobs
	Stimmen-ID: ListFraudsters
	Stimmen-ID: ListSpeaker EnrollmentJobs
	Stimmen-ID: ListSpeakers
	Stimmen-ID: ListWatchlists

Servicepräfix	Aktionen
	voiceid: Sprecher OptOut Stimmen-ID: StartFraudster RegistrationJob Stimmen-ID: StartSpeaker EnrollmentJob Stimmen-ID: UpdateDomain Stimmen-ID: UpdateWatchlist

Servicepräfix	Aktionen
vpc-lattice	vpc-Gitter: CreateAccess LogSubscription vpc-Gitter: CreateListener vpc-Gitter: CreateRule vpc-Gitter: CreateService vpc-lattice: Netzwerk CreateService vpc-lattice: Verband CreateService NetworkService vpc-lattice: Verband CreateService NetworkVpc vpc-lattice: Gruppe CreateTarget vpc-lattice: DeleteAccess LogSubscription vpc-lattice: Richtlinie DeleteAuth vpc-lattice: DeleteListener vpc-lattice: Richtlinie DeleteResource vpc-lattice: DeleteRule vpc-Gitter: DeleteService vpc-lattice: Netzwerk DeleteService vpc-lattice: Verband DeleteService NetworkService vpc-lattice: Verband DeleteService NetworkVpc vpc-lattice: Gruppe DeleteTarget vpc-lattice: DeregisterTargets vpc-Gitter: GetAccess LogSubscription vpc-lattice: Richtlinie GetAuth

Servicepräfix	Aktionen
	vpc-lattice: GetListener vpc-lattice: Richtlinie GetResource vpc-lattice: GetRule vpc-Gitter: GetService vpc-lattice: Netzwerk GetService vpc-lattice: Verband GetService NetworkService vpc-lattice: Verband GetService NetworkVpc vpc-lattice: Gruppe GetTarget vpc-lattice: ListAccess LogSubscriptions vpc-Gitter: ListListeners vpc-Gitter: ListRules vpc-lattice: Netzwerke ListService vpc-lattice: Verbände ListService NetworkService vpc-lattice: Verbände ListService NetworkVpc vpc-lattice: ListServices vpc-lattice: Gruppen ListTarget vpc-lattice: ListTargets vpc-lattice: Richtlinie PutAuth vpc-lattice: Richtlinie PutResource vpc-lattice: RegisterTargets vpc-Gitter: UpdateAccess LogSubscription

Servicepräfix	Aktionen
	vpc-Gitter: UpdateListener vpc-Gitter: UpdateRule vpc-Gitter: UpdateService vpc-lattice: Netzwerk UpdateService vpc-lattice: Verband UpdateService NetworkVpc vpc-lattice: Gruppe UpdateTarget

Servicepräfix	Aktionen
wafv2	wafv2: ACL AssociateWeb
	WAFv2: CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	WAF 2: CreateRegex PatternSet
	wafv2: Gruppe CreateRule
	wafv2: ACL CreateWeb
	WAFv2: API-Schlüssel löschen
	wafv2: Gruppen DeleteFirewall ManagerRule
	wafv2:DeleteIPSet
	wafv2: Konfiguration DeleteLogging
	wafv2: Richtlinie DeletePermission
	wafv2: DeleteRegex PatternSet
	wafv2: Gruppe DeleteRule
	wafv2: ACL DeleteWeb
	WAFv2: DescribeAll ManagedProducts
	wafv2: Anbieter DescribeManaged ProductsBy
	wafv2: DescribeManaged RuleGroup
	wafv2: ACL DisassociateWeb
	wafv2: URL GenerateMobile SdkRelease
	wafv2: API-Schlüssel GetDecrypted

Servicepräfix	Aktionen
	wafv2:GetIPSet
	wafv2: Konfiguration GetLogging
	wafv2: GetManaged RuleSet
	WAF 2: GetMobile SdkRelease
	wafv2: Richtlinie GetPermission
	wafv2: GetRate BasedStatement ManagedKeys
	WAF 2: GetRegex PatternSet
	wafv2: Gruppe GetRule
	wafv2: Anfragen GetSampled
	wafv2: ACL GetWeb ForResource
	wafv2:ListAPIKeys
	wafv2: Gruppen ListAvailable ManagedRule
	wafv2: ListAvailable ManagedRule GroupVersions
	wafv2:ListIPSets
	wafv2: Konfigurationen ListLogging
	wafv2: ListManaged RuleSets
	WAF 2: ListMobile SdkReleases
	WAF 2: ListRegex PatternSets
	wafv2: ACL ListResources ForWeb
	wafv2: Gruppen ListRule
	wafv2: ACLs ListWeb

Servicepräfix	Aktionen
	wafv2: Konfiguration PutLogging wafv2: Versionen PutManaged RuleSet wafv2: Richtlinie PutPermission wafv2:UpdateIPSet wafv2: Datum UpdateManaged RuleSet VersionExpiry wafv2: UpdateRegex PatternSet wafv2: Gruppe UpdateRule wafv2: ACL UpdateWeb

Servicepräfix	Aktionen
wellarchitected	gut konzipiert: AssociateLenses gut gestaltet: AssociateProfiles wellarchitected: Teilen CreateLens wellarchitected: Version CreateLens gut strukturiert: CreateMilestone gut gestaltet: CreateProfile wellarchitected: Teilen CreateProfile wellarchitected: Vorlage CreateReview gut strukturiert: CreateWorkload wellarchitected: Teilen CreateWorkload wellarchitected: DeleteLens wellarchitected: Teilen DeleteLens wellarchitected: DeleteProfile wellarchitected: Teilen DeleteProfile wellarchitected: Vorlage DeleteReview wellarchitected: Teilen DeleteTemplate wellarchitected: DeleteWorkload wellarchitected: Teilen DeleteWorkload wellarchitected: DisassociateLenses gut gestaltet: DisassociateProfiles gut gestaltet: ExportLens

Servicepräfix	Aktionen
	gut gestaltet: GetAnswer wellarchitected: Bericht GetConsolidated wellarchitected: Einstellungen GetGlobal gut strukturiert: GetLens wellarchitected: Rezension GetLens gut strukturiert: GetLens ReviewReport gut gestaltet: GetLens VersionDifference gut gestaltet: GetMilestone gut gestaltet: GetProfile wellarchitected: Vorlage GetProfile wellarchitected: Vorlage GetReview gut strukturiert: GetReview TemplateAnswer wellarchitected: Rückblick GetReview TemplateLens gut strukturiert: GetWorkload gut gestaltet: ImportLens gut gestaltet: ListAnswers gut gestaltet: Einzelheiten ListCheck wellarchitected: Zusammenfassungen ListCheck gut strukturiert: ListLenses gut gestaltet: ListLens ReviewImprovements wellarchitected: Bewertungen ListLens

Servicepräfix	Aktionen
	wellarchitected: Aktien ListLens gut strukturiert: ListMilestones gut gestaltet: ListNotifications wellarchitected: Benachrichtigungen ListProfile gut strukturiert: ListProfiles wellarchitected: Aktien ListProfile gut strukturiert: ListReview TemplateAnswers wellarchitected: Vorlagen ListReview wellarchitected: Einladungen ListShare wellarchitected: Aktien ListTemplate gut strukturiert: ListWorkloads wellarchitected: Aktien ListWorkload gut strukturiert: UpdateAnswer wellarchitected: Einstellungen UpdateGlobal gut strukturiert: UpdateIntegration wellarchitected: Rezension UpdateLens gut strukturiert: UpdateProfile wellarchitected: Vorlage UpdateReview wellarchitected: Rezension UpdateReview TemplateLens wellarchitected: Einladung UpdateShare gut gestaltet: UpdateWorkload

Servicepräfix	Aktionen
	wellarchitected: Teilen UpdateWorkload wellarchitected: Rezension UpgradeLens wellarchitected: Version UpgradeProfile wellarchitected: Rezension UpgradeReview TemplateLens

Servicepräfix	Aktionen
wisdom	Weisheit: CreateAssistant
	Weisheit: CreateAssistant Assoziation
	Weisheit: CreateContent
	Weisheit: CreateKnowledge Basis
	Weisheit: CreateQuick Antwort
	Weisheit: CreateSession
	Weisheit: DeleteAssistant
	Weisheit: DeleteAssistant Assoziation
	Weisheit: DeleteContent
	Weisheit: DeleteImport Job
	Weisheit: DeleteKnowledge Basis
	Weisheit: DeleteQuick Antwort
	Weisheit: GetAssistant
	Weisheit: GetAssistant Assoziation
	Weisheit: GetContent
	Weisheit: GetContent Zusammenfassung
	Weisheit: GetImport Job
	Weisheit: GetKnowledge Basis
	Weisheit: GetRecommendations
	Weisheit: GetSession
	Weisheit: ListAssistant Assoziationen

Servicepräfix	Aktionen
	Weisheit: ListAssistants
	Weisheit: ListContents
	Weisheit: ListImport Jobs
	Weisheit: ListKnowledge Grundlagen
	Weisheit: ListQuick Antworten
	Weisheit: NotifyRecommendations Empfangen
	Weisheit: QueryAssistant
	Weisheit: RemoveKnowledge BaseTemplate Uri
	Weisheit: SearchContent
	Weisheit: SearchQuick Antworten
	Weisheit: SearchSessions
	Weisheit: StartContent Hochladen
	Weisheit: StartImport Job
	Weisheit: UpdateContent
	Weisheit: UpdateKnowledge BaseTemplate Uri
	Weisheit: UpdateQuick Antwort

Servicepräfix	Aktionen
worklink	Arbeitslink: AssociateDomain Arbeitslink: AssociateWebsite AuthorizationProvider Arbeitslink: AssociateWebsite CertificateAuthority Arbeitslink: CreateFleet Arbeitslink: DeleteFleet Arbeitslink: DescribeAudit StreamConfiguration Arbeitslink: DescribeCompany NetworkConfiguration Arbeitslink: DescribeDevice Arbeitslink: DescribeDevice PolicyConfiguration Arbeitslink: DescribeDomain worklink: Metadaten DescribeFleet Arbeitslink: DescribeIdentity ProviderConfiguration Arbeitslink: DescribeWebsite CertificateAuthority Arbeitslink: DisassociateDomain Arbeitslink: DisassociateWebsite AuthorizationProvider Arbeitslink: DisassociateWebsite CertificateAuthority Arbeitslink: ListDevices Arbeitslink: ListDomains Arbeitslink: ListFleets Arbeitslink: ListWebsite AuthorizationProviders Arbeitslink: ListWebsite CertificateAuthorities

Servicepräfix	Aktionen
	worklink: Zugriff RestoreDomain worklink: Zugriff RevokeDomain worklink: Benutzer SignOut Arbeitslink: UpdateAudit StreamConfiguration Arbeitslink: UpdateCompany NetworkConfiguration Arbeitslink: UpdateDevice PolicyConfiguration worklink: Metadaten UpdateDomain worklink: Metadaten UpdateFleet Arbeitslink: UpdateIdentity ProviderConfiguration

Servicepräfix	Aktionen
Workspaces	Arbeitsbereiche: AcceptAccount LinkInvitation
	Arbeitsbereiche: Alias AssociateConnection
	Arbeitsbereiche: Gruppen AssociateIp
	Arbeitsbereiche: Anwendung AssociateWorkspace
	Arbeitsbereiche: Bild CopyWorkspace
	Arbeitsbereiche: In CreateConnect ClientAdd
	Arbeitsbereiche: Alias CreateConnection
	Arbeitsbereiche: Gruppe CreateIp
	Arbeitsbereiche: Arbeitsbereiche CreateStandby
	Arbeitsbereiche: CreateUpdated WorkspaceImage
	Arbeitsbereiche: Paket CreateWorkspace
	Arbeitsbereiche: Bild CreateWorkspace
	Arbeitsbereiche: CreateWorkspaces
	Arbeitsbereiche: Branding DeleteClient
	Arbeitsbereiche: In DeleteConnect ClientAdd
	Arbeitsbereiche: Alias DeleteConnection
	Arbeitsbereiche: Gruppe DeleteIp
	Arbeitsbereiche: Paket DeleteWorkspace
	Arbeitsbereiche: Bild DeleteWorkspace
	Arbeitsbereiche: Anwendungen DeployWorkspace
	Arbeitsbereiche: Verzeichnis DeregisterWorkspace

Servicepräfix	Aktionen
	Arbeitsbereiche: DescribeAccount
	Arbeitsbereiche: Änderungen DescribeAccount
	Arbeitsbereiche: Assoziationen DescribeApplication
	Arbeitsbereiche: DescribeApplications
	Arbeitsbereiche: Verbände DescribeBundle
	Arbeitsbereiche: Branding DescribeClient
	Arbeitsbereiche: Eigenschaften DescribeClient
	Arbeitsbereiche: Ins DescribeConnect ClientAdd
	Arbeitsbereiche: Aliase DescribeConnection
	Arbeitsbereiche: DescribeConnection AliasPermissions
	Arbeitsbereiche: Verbände DescribeImage
	Arbeitsbereiche: Gruppen DescribeIp
	Arbeitsbereiche: Verbände DescribeWorkspace
	Arbeitsbereiche: Pakete DescribeWorkspace
	Arbeitsbereiche: Verzeichnisse DescribeWorkspace
	Arbeitsbereiche: DescribeWorkspace ImagePermissions
	Arbeitsbereiche: DescribeWorkspaces
	Arbeitsbereiche: DescribeWorkspaces ConnectionStatus
	Arbeitsbereiche: Schnappschüsse DescribeWorkspace
	Arbeitsbereiche: Alias DisassociateConnection
	Arbeitsbereiche: Gruppen DisassociateIp

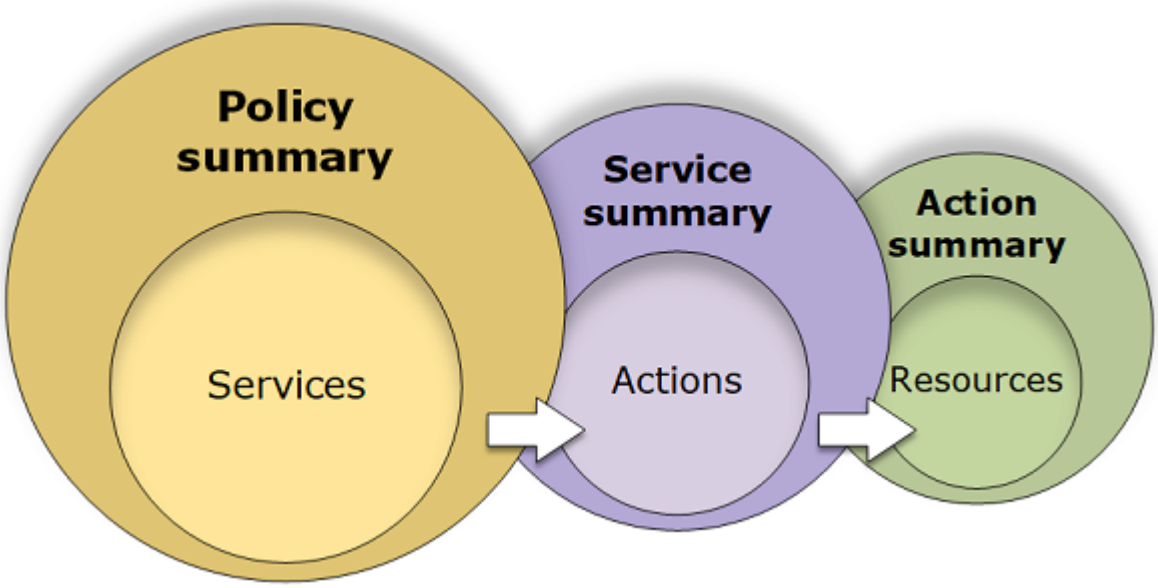
Servicepräfix	Aktionen
	Arbeitsbereiche: Anwendung DisassociateWorkspace
	Arbeitsbereiche: Link GetAccount
	Arbeitsbereiche: Branding ImportClient
	Arbeitsbereiche: Bild ImportWorkspace
	Arbeitsbereiche: Links ListAccount
	Arbeitsbereiche: Bereiche ListAvailable ManagementCidr
	Arbeitsbereiche: MigrateWorkspace
	Arbeitsbereiche: ModifyAccount
	Arbeitsbereiche: Eigenschaften ModifyCertificate BasedAuth
	Arbeitsbereiche: Eigenschaften ModifyClient
	Arbeitsbereiche: Eigenschaften ModifySaml
	Arbeitsbereiche: Berechtigungen ModifySelfservice
	Arbeitsbereiche: ModifyWorkspace AccessProperties
	Arbeitsbereiche: ModifyWorkspace CreationProperties
	Arbeitsbereiche: Eigenschaften ModifyWorkspace
	Arbeitsbereiche: Bundesstaat ModifyWorkspace
	Arbeitsbereiche: RebootWorkspaces
	Arbeitsbereiche: RebuildWorkspaces
	Arbeitsbereiche: Verzeichnis RegisterWorkspace
	Arbeitsbereiche: RejectAccount LinkInvitation
	Arbeitsbereiche: RestoreWorkspace

Servicepräfix	Aktionen
	Arbeitsbereiche: StartWorkspaces
	Arbeitsbereiche: StopWorkspaces
	Arbeitsbereiche: TerminateWorkspaces
	Arbeitsbereiche: In UpdateConnect ClientAdd
	Arbeitsbereiche: UpdateConnection AliasPermission
	Arbeitsbereiche: Paket UpdateWorkspace
	Arbeitsbereiche: UpdateWorkspace ImagePermission

Servicepräfix	Aktionen
xray	röntgen: CreateGroup xray: Regel CreateSampling röntgen: DeleteGroup xray: Richtlinie DeleteResource xray: Regel DeleteSampling xray: Config GetEncryption Röntgen: GetGroup röntgen: GetGroups röntgen: GetInsight xray: Ereignisse GetInsight röntgen: GetInsight ImpactGraph xray: Zusammenfassungen GetInsight xray: Regeln GetSampling xray: Richtlinien ListResource xray: Config PutEncryption xray: Richtlinie PutResource Röntgen: UpdateGroup xray: Regel UpdateSampling

Grundlegendes zu von Richtlinien gewährten Berechtigungen

Die IAM-Konsole enthält Tabellen mit Richtlinienübersichten, die die Zugriffsebene, Ressourcen und Bedingungen aufführt, die für die einzelnen Services in einer Richtlinie zugelassen oder verweigert werden. Richtlinien werden in drei Tabellen zusammengefasst: Richtlinienübersicht, Serviceübersicht und Aktionsübersicht. Die Tabelle mit der Richtlinienübersicht enthält eine Liste von Services. Wählen Sie dort einen Service aus, um die Serviceübersicht anzuzeigen. Diese Übersichtstabelle enthält eine Liste der Aktionen und dazugehörigen Berechtigungen für den ausgewählten Service. Sie können eine Aktion in dieser Tabelle auswählen, um die Aktionsübersicht anzuzeigen. Diese Tabelle enthält eine Liste der Ressourcen und Bedingungen für die gewählte Aktion.



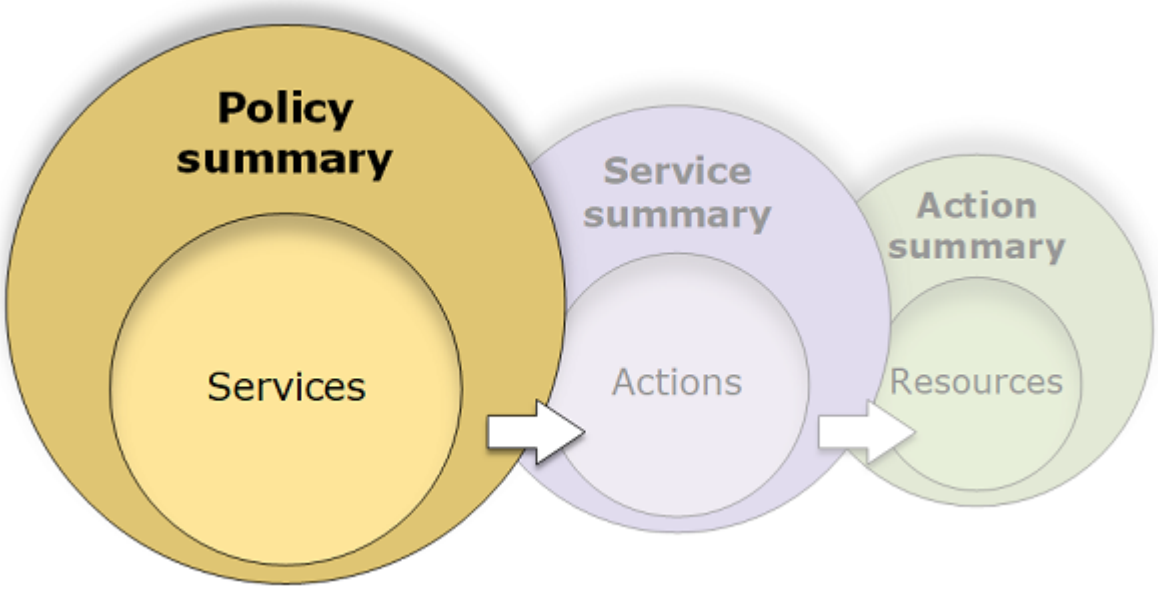
Sie können auf der Seite Users (Benutzer) oder Roles (Rollen) Richtlinienübersichten für alle Richtlinien (verwaltete Richtlinien und Inlinerichtlinien) anzeigen, die diesem Benutzer angefügt sind. Auf der Seite Policies (Richtlinien) können Sie Übersichten für alle verwalteten Richtlinien anzeigen. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien, AWS verwaltete Richtlinien für Jobfunktionen und vom Kunden verwaltete Richtlinien. Sie können auf der Seite Policies (Richtlinien) Übersichten für diese Richtlinien anzeigen, unabhängig davon, ob sie an einen Benutzer oder eine andere IAM-Identität angefügt sind.

Sie können die Informationen in den Richtlinienübersichten verwenden, um die Berechtigungen, die von Ihrer Richtlinie zugelassen oder verweigert werden, umfassend zu verstehen. Richtlinienübersichten können Ihnen die Problembehandlung und Korrektur von Richtlinien erleichtern, die nicht die Berechtigungen bereitstellen, die Sie erwarten.

Themen

Richtlinienübersicht (Liste der Services)

Richtlinien werden in drei Tabellen zusammengefasst: Richtlinienübersicht, Serviceübersicht und Aktionsübersicht. In der Tabelle Richtlinienübersicht werden die Services und die Übersichten der für die ausgewählte Richtlinie definierten Berechtigungen aufgelistet.



Die Tabelle der Richtlinienübersicht ist in eine oder mehrere Abschnitte gruppiert: Uncategorized services (Nicht kategorisierte Services), Explicit deny (Explizite Zugriffsverweigerung) und Allow (Erlauben). Wenn die Richtlinie einen Service enthält, den IAM nicht erkennt, ist der Service im Abschnitt Uncategorized services der Tabelle enthalten. Wenn IAM den Service erkennt, ist er je nach Auswirkung der Richtlinie (oder) im Abschnitt Explicit deny oder `Deny`Allow`Allow` der Tabelle enthalten.

Anzeigen von Richtlinienübersichten

Sie können die Zusammenfassungen aller Richtlinien einsehen, die an einen Benutzer angehängt sind, indem Sie auf der Seite mit den Benutzerdetails auf der Registerkarte Permissions

(Berechtigungen) den Richtliniennamen auswählen. Sie können die Zusammenfassungen aller Richtlinien einsehen, die an eine Rolle angehängt sind, indem Sie auf der Seite mit den Rollendetails auf der Registerkarte Permissions (Berechtigungen) den Richtliniennamen auswählen. Sie können die Richtlinienübersicht für alle verwalteten Richtlinien auf der Seite Policies (Richtlinien) einsehen. Wenn Ihre Richtlinie keine Richtlinienübersicht enthält, finden Sie unter Übersicht fehlender Richtlinien Informationen zu den Gründen dafür.

So zeigen Sie die Richtlinienübersicht auf der Seite Policies (Richtlinien) an

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 Policies.
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie anzeigen möchten.
4. Wählen Sie auf der Seite Policy details (Richtliniendetails) der Richtlinie die Registerkarte Permissions (Berechtigungen) aus, um die Richtlinienübersicht anzuzeigen.

So zeigen Sie die Übersicht für eine an einen Benutzer angefügte Richtlinie an

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 Users (Benutzer).
3. Wählen Sie in der Benutzerliste den Namen des Benutzers aus, dessen Richtlinie Sie anzeigen möchten.
4. Wählen Sie auf der Seite Summary (Übersicht) des Benutzers die Registerkarte Permissions (Berechtigungen) aus, um die Liste der Richtlinien anzuzeigen, die direkt oder aus einer Gruppe an den Benutzer angefügt werden.
5. Erweitern Sie in der Richtlinientabelle des Benutzers die Zeile der anzuzeigenden Richtlinie.

So zeigen Sie die Übersicht für eine an eine Rolle angefügte Richtlinie an

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 Rollen aus.
3. Wählen Sie in der Rollenliste den Namen der Rolle aus, deren Richtlinie Sie anzeigen möchten.

4. Wählen Sie auf der Seite Summary (Zusammenfassung) der Rolle die Registerkarte Permissions (Berechtigungen) aus, um die Liste der Richtlinien anzuzeigen, die der Rolle zugeordnet sind.
5. Erweitern Sie in der Richtlinientabelle der Rolle die Zeile der anzuzeigenden Richtlinie.

Bearbeiten von Richtlinien zur Behebung von Warnungen

Beim Anzeigen einer Richtlinienübersicht erkennen Sie möglicherweise einen Rechtschreibfehler oder Sie stellen fest, dass die Richtlinie nicht die erwarteten Berechtigungen hat. Sie können eine Richtlinienübersicht nicht direkt bearbeiten. Sie können jedoch eine individuell verwaltete Richtlinie mit dem visuellen Richtlinieneditor bearbeiten, der viele der Fehler und Warnungen erfasst, die in den Richtlinienzusammenfassungsberichten angezeigt werden. Sie können dann die Änderungen in der Zusammenfassung einsehen, um zu prüfen, ob Sie alle Probleme behoben haben. Informationen zum Bearbeiten einer Inline-Richtlinie finden Sie unter the section called “Bearbeiten von IAM-Richtlinien”. Sie können AWS verwaltete Richtlinien nicht bearbeiten.

So bearbeiten Sie eine Richtlinie für die Richtlinienübersicht mit der Visual-Option

1. Öffnen Sie die Richtlinienübersicht wie in den vorherigen Verfahren erläutert.
2. Wählen Sie Bearbeiten aus.

   Wenn Sie die Seite Users (Benutzer) geöffnet haben und eine vom Kunden verwaltete Richtlinie bearbeiten möchten, die mit diesem Benutzer verknüpft ist, werden Sie zur Seite Policies (Richtlinien) weitergeleitet. Die vom Kunden verwalteten Richtlinien können nur auf der Seite Policies (Richtlinien) bearbeitet werden.
3. Wählen Sie die Visual-Option aus, um die bearbeitungsfähige, visuelle Darstellung Ihrer Richtlinie anzuzeigen. IAM kann Ihre Richtlinie umstrukturieren, um sie für den visuellen Editor zu optimieren und Ihnen das Auffinden und Beheben von Problemen zu erleichtern. Die Warnungen und Fehlermeldungen auf der Seite können Ihnen helfen, Probleme mit Ihrer Richtlinie zu beheben. Weitere Informationen zur Restrukturierung von Richtlinien durch IAM finden Sie unter Umstrukturierung einer Richtlinie.
4. Bearbeiten Sie die Richtlinien und wählen Sie Next (Weiter) aus, um die übernommenen Änderungen in der Richtlinienübersicht einzusehen. Wenn Sie noch immer ein Problem sehen, wählen Sie Previous (Zurück) aus, um zur Bearbeitungsseite zurückzukehren.
5. Wählen Sie Änderungen speichern aus, um Ihre Änderungen zu speichern.

So bearbeiten Sie eine Richtlinie für die Richtlinienübersicht mit der JSON-Option

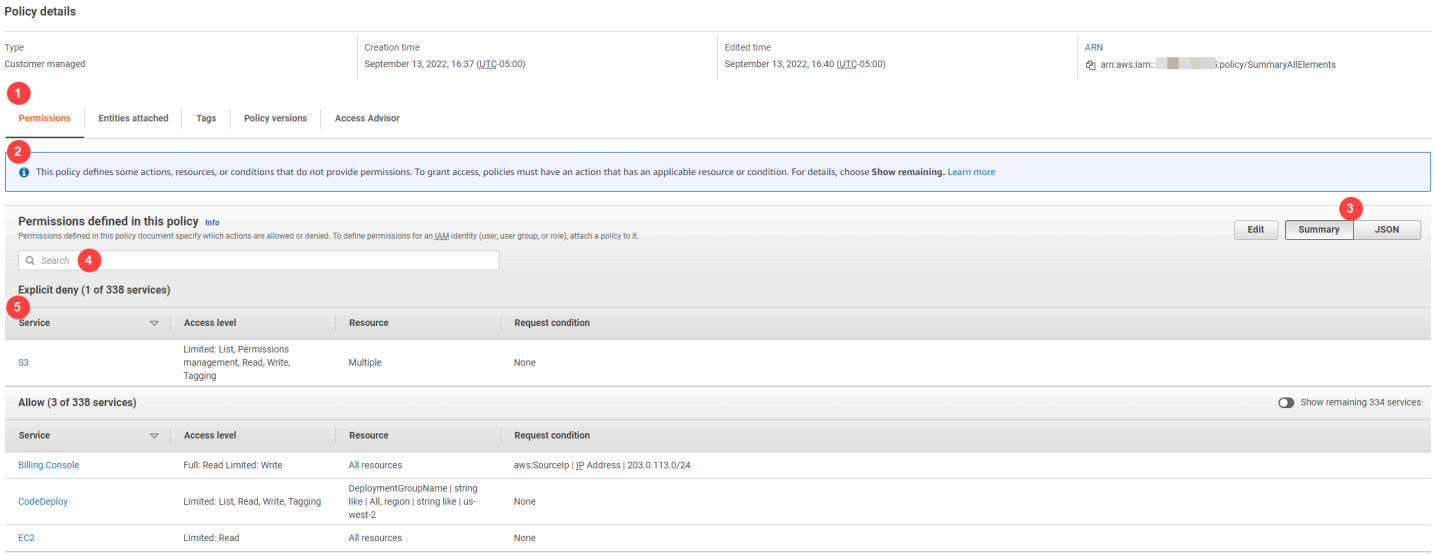
1. Öffnen Sie die Richtlinienübersicht wie in den vorherigen Verfahren erläutert.
2. Verwenden Sie die Schaltflächen Summary (Übersicht) und JSON, um die Richtlinienübersicht und dem JSON-Richtliniendokument zu vergleichen. Anhand dieser Informationen können Sie bestimmen, welche Zeilen im Richtliniendokument Sie ändern möchten.
3. Wählen Sie Edit (Bearbeiten) und anschließend die JSON-Option aus, um das JSON-Richtliniendokument zu bearbeiten.

   > **Note**
   >
   > Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder in der Editoroption Visual Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

   Wenn Sie die Seite Users (Benutzer) geöffnet haben und eine vom Kunden verwaltete Richtlinie bearbeiten möchten, die mit diesem Benutzer verknüpft ist, werden Sie zur Seite Policies (Richtlinien) weitergeleitet. Die vom Kunden verwalteten Richtlinien können nur auf der Seite Policies (Richtlinien) bearbeitet werden.
4. Bearbeiten Sie Ihre Richtlinie. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter. Wenn Sie noch immer ein Problem sehen, wählen Sie Previous (Zurück) aus, um zur Bearbeitungsseite zurückzukehren.
5. Wählen Sie Änderungen speichern aus, um Ihre Änderungen zu speichern.

Elemente einer Richtlinienübersicht

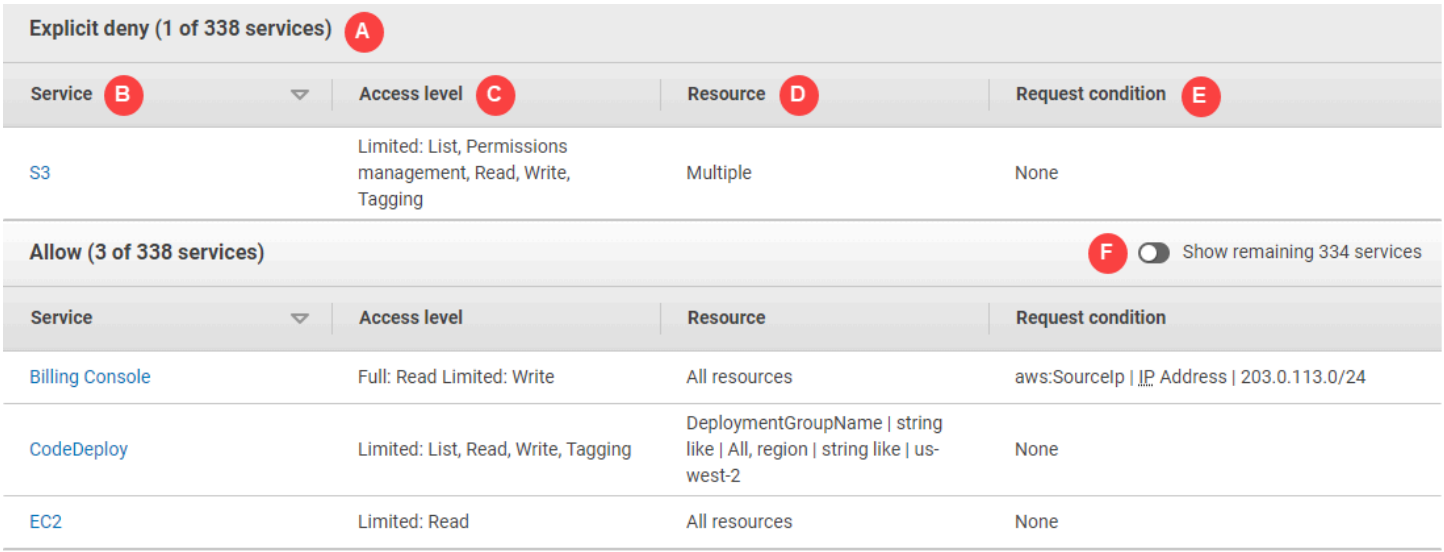
Im folgenden Beispiel einer Seite mit Richtliniendetails handelt es sich bei der SummaryAllElementsRichtlinie um eine verwaltete Richtlinie (vom Kunden verwaltete Richtlinie), die direkt an den Benutzer angehängt ist. Diese Richtlinie wird erweitert, um die Richtlinienübersicht einzublenden.



In der vorhergehenden Abbildung wird die Richtlinienübersicht auf der Seite Policies (Richtlinien) angezeigt:

1. Auf der Registerkarte Permissions (Berechtigungen) werden die in der Richtlinie definierten Berechtigungen angezeigt.
2. Wenn die Richtlinie nicht Berechtigungen für alle Aktionen, Ressourcen und Bedingungen erteilt, die in der Richtlinie definiert sind, wird oben auf der Seite ein Warn- oder Fehlerbanner angezeigt. Die Richtlinienübersicht enthält dann Details zu dem Problem. Informationen dazu, wie Richtlinienübersichten dabei helfen, Probleme mit den Berechtigungen, die durch Ihre Richtlinie erteilt werden, zu verstehen und zu beheben, finden Sie unter the section called “Meine Richtlinie erteilt nicht die erwarteten Berechtigungen”.
3. Verwenden Sie die Schaltflächen Summary (Übersicht) und JSON, um zwischen der Richtlinienübersicht und dem JSON-Richtliniendokument zu wechseln.
4. Verwenden Sie das Feld Search (Suche), um die Liste der Services zu kürzen und nach einem bestimmten Service zu suchen.
5. In der erweiterten Ansicht werden zusätzliche Details der SummaryAllElementsRichtlinie angezeigt.

Die folgende Tabelle mit der Zusammenfassung der Richtlinie zeigt die erweiterte SummaryAllElementsRichtlinie auf der Seite mit den Richtliniendetails.



Explicit deny (1 of 338 services) A

Service B	Access level C	Resource D	Request condition E
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (3 of 338 services) F Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeDeploy	Limited: List, Read, Write, Tagging	DeploymentGroupName | string like | All, region | string like | us-west-2	None
EC2	Limited: Read	All resources	None

In der vorhergehenden Abbildung wird die Richtlinienübersicht auf der Seite Policies (Richtlinien) angezeigt:

A. Solche Services, die IAM erkennt, werden entsprechend in Gruppen eingeteilt, je nachdem, ob die Richtlinie die Verwendung des Services zulässt oder explizit verweigert. In diesem Beispiel umfasst die Richtlinie eine `Deny` Erklärung für den Amazon S3 S3-Service und `Allow` Kontoauszüge für die Billing CodeDeploy - und Amazon EC2-Services.

B. Service – In dieser Spalte werden die in der Richtlinie definierten Services aufgelistet und die Details für den jeweiligen Service angegeben. Jeder Servicename in der Richtlinienübersichtstabelle ist als Link zur Tabelle Serviceübersicht formatiert, die unter Serviceübersicht (Liste der Aktionen) erläutert wird. In diesem Beispiel werden Berechtigungen für die Amazon S3- CodeDeploy, Billing- und Amazon EC2-Services definiert.

C. Access Level (Zugriffsebene) – In dieser Spalte ist angegeben, ob die Aktionen in jeder Zugriffsebene (`List`, `Read`, `Write`, `Permission Management` und `Tagging`) über die in der Richtlinie definierten Berechtigungen `Full` oder `Limited` verfügen. Weitere Informationen und Beispiele für die Zugriffsebenenübersicht finden Sie unter Zugriffsebenen in Richtlinienzusammenfassungen verstehen.

- Full access (Voller Zugriff) – Dieser Eintrag gibt an, dass der Service Zugriff auf alle Aktionen in den für den Service verfügbaren vier Zugriffsebenen hat.
- Wenn der Eintrag nicht Full access (Voller Zugriff) enthält, hat der Service Zugriff nur auf einige, aber nicht auf alle Aktionen des Services. Der Zugriff wird dann durch die folgenden Beschreibungen für jede der Zugriffsebenenklassifizierungen (`List`, `Read`, `Write`, `Permission Management` und `Tagging`) definiert:

Full (Voll): Die Richtlinie bietet Zugriff auf alle Aktionen in jeder der aufgeführten Zugriffsebenenklassifizierungen. In diesem Beispiel bietet die Richtlinie Zugriff auf alle `Read`-Fakturierungsaktionen.

Limited (Beschränkt): Die Richtlinie bietet Zugriff auf eine oder mehrere Aktionen in jeder der aufgeführten Zugriffsebenenklassifizierungen, jedoch nicht auf alle Aktionen. In diesem Beispiel bietet die Richtlinie Zugriff auf einige `Write`-Fakturierungsaktionen.

D. Resource (Ressource) – In dieser Spalte werden die Ressourcen aufgelistet, die die Richtlinie für den jeweiligen Service definiert.

- Multiple (Mehrfach) – Die Richtlinie enthält mehr als eine Ressource in dem Service, jedoch nicht alle Ressourcen. In diesem Beispiel wird der Zugriff auf mehr als eine Amazon S3-Ressource explizit verweigert.
- All resources (Alle Ressourcen) – Die Richtlinie ist für alle Ressourcen in dem Service definiert. In diesem Beispiel ermöglicht es die Richtlinie, die angegebenen Aktionen für alle Fakturierungsressourcen auszuführen.
- Resource text – Die Richtlinie enthält eine Ressource in dem Service. In diesem Beispiel sind die aufgelisteten Aktionen nur für die `DeploymentGroupName` CodeDeploy Ressource zulässig. Je nach den Informationen, die der Service für IAM bereitstellt, wird eventuell ein ARN oder der definierte Ressourcentyp angezeigt.

> **Note**
>
> Diese Spalte kann eine Ressource aus einem anderen Service enthalten. Wenn die Richtlinienanweisung, die die Ressource umfasst, nicht über die Aktionen und Ressourcen aus demselben Service verfügt, enthält Ihre Richtlinie nicht übereinstimmende Ressourcen. Sie werden nicht von IAM über nicht übereinstimmende Ressourcen gewarnt, wenn Sie eine Richtlinie erstellen oder in der Richtlinienübersicht anzeigen. Wenn diese Spalte eine nicht übereinstimmende Ressource enthält, sollten Sie Ihre Richtlinien auf Fehler überprüfen. Um Ihre Richtlinien besser zu verstehen, testen Sie sie stets mit dem Richtliniensimulator.

E. Request condition (Anforderungsbedingung) – In dieser Spalte wird angegeben, ob die der Ressource zugeordneten Services oder Aktionen Bedingungen unterliegen.

- None (Keine) – Die Richtlinie enthält keine Bedingungen für den Service. In diesem Beispiel werden keine Bedingungen auf die verweigerten Aktionen im Amazon S3-Dienst angewendet.

- Condition text – Die Richtlinie enthält eine Bedingung für den Service. In diesem Beispiel sind die aufgeführten Billing-Aktionen nur dann zulässig, wenn die IP-Adresse der Quelle mit `203.0.113.0/24` übereinstimmt.
- Multiple (Mehrfach) – Die Richtlinie enthält mehrere Bedingungen für den Service. Zum Anzeigen der jeweiligen Bedingungen für die Richtlinie wählen Sie JSON aus, um das Richtliniendokument einzusehen.

F. Show remaining services (Die restlichen Services anzeigen) – Klicken Sie auf diese Schaltfläche, um die Tabelle zu erweitern und auch diejenigen Services anzuzeigen, die durch die Richtlinie nicht definiert sind. Diese Services werden in dieser Richtlinie implizit verweigert (oder standardmäßig verweigert). Eine Anweisung in einer anderen Richtlinie kann möglicherweise jedoch weiterhin die Verwendung des Services zulassen oder explizit verweigern. In der Richtlinienübersicht sind die Berechtigungen einer einzelnen Richtlinie zusammengefasst. Informationen darüber, wie der AWS Dienst entscheidet, ob eine bestimmte Anfrage zugelassen oder abgelehnt werden soll, finden Sie unterAuswertungslogik für Richtlinien.

Wenn eine Richtlinie oder ein Element in der Richtlinie keine Berechtigungen erteilt, zeigt IAM zusätzliche Warnungen und Informationen in der Richtlinienübersicht an. Die folgende Tabelle mit der Zusammenfassung der Richtlinien zeigt die erweiterten Dienste „Verbleibende Dienste anzeigen“ auf der Seite mit den SummaryAllElementsRichtliniendetails sowie die möglichen Warnungen.

Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

In der Abbildung oben sehen Sie alle Services, die definierten Aktionen, Ressourcen oder Bedingungen ohne Berechtigungen enthalten:

a. Resource warnings (Ressourcenwarnungen) – Für Services, die keine Berechtigungen für alle enthaltenen Aktionen oder Ressourcen bereitstellen, wird eine der folgenden Warnungen in der Spalte Resource (Ressource) der Tabelle angezeigt:

   - 

     No resources are defined (Keine Ressourcen definiert). – Das bedeutet, dass der Service Aktionen definiert hat, aber die Richtlinie keine unterstützten Ressourcen enthält.

   - 

     One or more actions do not have an applicable resource (Eine oder mehrere Aktionen verfügen über keine geeignete Ressource). – Das bedeutet, dass der Service Aktionen definiert hat, aber dass einige dieser Aktionen nicht über eine unterstützte Ressource verfügen.

   - 

     Eine oder mehrere Ressourcen haben keine anwendbare Aktion. – Das bedeutet, dass der Service Ressourcen definiert hat, aber dass einige dieser Ressourcen nicht über eine unterstützte Aktion verfügen.

   Wenn ein Service sowohl Aktionen enthält, die keine zugehörigen Ressourcen besitzen, als auch Ressourcen, die zugehörigen Ressourcen besitzen, wird nur die Warnung One or more resources do not have an applicable action (Eine oder mehrere Ressourcen haben keine anwendbare Aktion) angezeigt. Der Grund hierfür ist, dass bei der Anzeige der Serviceübersicht für den Service, Ressourcen, die auf keine Aktion angewendet werden, nicht angezeigt werden. Für die Aktion `ListAllMyBuckets` enthält diese Richtlinie die letzte Warnung, da die Aktion keine Berechtigungen auf Ressourcenebene und nicht den Bedingungsschlüssel `s3:x-amz-acl` unterstützt. Wenn Sie entweder das Problem mit der Ressource oder der Bedingung beheben, wird nur das verbleibende Problem in einer detaillierten Warnung angezeigt.

b. Request condition warnings (Anfordern von Bedingungswarnungen) – Für Services, die keine Berechtigungen für alle enthaltenen Bedingungen bereitstellen, wird eine der folgenden Warnungen in der Spalte Request condition (Bedingung anfordern) der Tabelle angezeigt:

   - 

     One or more actions do not have an applicable condition (Eine oder mehrere Aktionen haben keine anwendbare Bedingung.). – Das bedeutet, dass der Service Aktionen definiert hat, aber dass einige dieser Aktionen nicht über eine unterstützte Bedingung verfügen.

- 

  One or more conditions do not have an applicable action (Eine oder mehrere Bedingungen haben keine anwendbare Aktion). – Das bedeutet, dass der Service Bedingungen definiert hat, aber dass einige dieser Bedingungen nicht über eine unterstützte Aktion verfügen.

c. Multiple |

One or more actions do not have an applicable resource (Eine oder mehrere Aktionen verfügen über keine geeignete Ressource). – Die `Deny`-Anweisung für Amazon S3 umfasst mehr als eine Ressource. Außerdem enthält sie mehrere Aktionen – einige Aktionen unterstützen die Ressourcen, andere wiederum nicht. Informationen zu dieser Richtlinie finden Sie unter: the section called “SummaryAllElements ein JSON-Richtliniendokument”. In diesem Fall enthält die Richtlinie alle Amazon S3-Aktionen. Es werden nur die Aktionen verweigert, die in einem Bucket oder Bucket-Objekt durchgeführt werden.

d.  No resources are defined – Der Service verfügt über definierte Aktionen, aber in die Richtlinie sind keine unterstützten Ressourcen enthalten, sodass der Service keine Berechtigungen bereitstellt. In diesem Fall umfasst die Richtlinie CodeCommit Aktionen, aber keine CodeCommit Ressourcen.

e. DeploymentGroupName | string like | All, region | string like | us-west-2 |

Eine oder mehrere Aktionen haben keine passende Ressource. – Der Service verfügt über eine definierte Aktion und mindestens eine weitere Aktion, die über keine unterstützende Ressource verfügt.

f. None |

One or more conditions do not have an applicable action. (Eine oder mehrere Bedingungen haben keine anwendbare Aktion). – Der Service verfügt über mindestens einen Bedingungsschlüssel, der über keine unterstützende Aktion verfügt.

SummaryAllElements ein JSON-Richtliniendokument

Die SummaryAllElementsRichtlinie ist nicht dafür vorgesehen, dass Sie sie verwenden, um Berechtigungen in Ihrem Konto zu definieren. Sie soll stattdessen veranschaulichen, welche Fehler und Warnungen beim Anzeigen einer Richtlinienübersicht auftreten können.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::developer_bucket",
                "arn:aws:s3:::developer_bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Zugriffsebenen in Richtlinienzusammenfassungen verstehen

AWS Zusammenfassung der Zugriffsebenen

Richtlinienübersichten enthalten eine Übersicht auf Zugriffsebene, in der die Aktionsberechtigungen beschrieben werden, die für jeden in der Richtlinie erwähnten Service definiert sind. Weitere Informationen zu Richtlinienübersichten finden Sie unter Grundlegendes zu von Richtlinien gewährten Berechtigungen. Übersichten über die Zugriffsebenen enthalten Angaben darüber, ob die Aktionen in jeder Zugriffsebene (`List`, `Read`, `Tagging`, `Write`, und `Permissions management`) die in der Richtlinie definierten Berechtigungen `Full` oder `Limited` haben. Informationen zur Zugriffsebenenklassifizierung, die jeder Aktion in einem Service zugewiesen ist, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Im nachfolgenden Beispiel werden die Zugriffsebenen beschrieben, die in einer Richtlinie für bestimmte Services gewährt werden. Beispiele mit vollständigen JSON-Richtliniendokumenten sowie deren Übersichten finden Sie unter Beispiele für Richtlinienübersichten.

Service	Zugriffsebene	Diese Richtlinie bietet Folgendes:
IAM	Vollzugriff	Zugriff auf alle Aktionen innerhalb des IAM-Services
CloudWatch	Full: List	Zugriff auf alle CloudWatch Aktionen in der `List` Zugriffsebene, aber kein Zugriff auf Aktionen mit der `Permissions management` Zugriffsebenenklassifizierung `ReadWrite`, oder.
Data Pipeline	Limited: List, Read	Zugriff auf mindestens eine, aber nicht auf alle AWS Data Pipeline Aktionen in der `Read` Zugriffsebene `List` und, aber nicht auf die `Permissions management` Aktionen `Write` Oder.
EC2	Full: List, Read Limited: Write	Zugriff auf alle Amazon EC2 `List`- und `Read`-Aktionen und Zugriff auf mindestens eine, aber nicht alle Amazon EC2 `Write`-Aktionen, aber

Service	Zugriffsebene	Diese Richtlinie bietet Folgendes:
		kein Zugriff auf Aktionen mit der `Permissions management` -Zugriffsstufeneinteilung.
S3	Limited: Read, Write, Permissions management	Zugriff auf mindestens eine, aber nicht alle Amazon S3-Aktionen mit `Read`, `Write` und `Permissions management`
CodeDeploy	(empty)	Unbekannter Zugriff, weil dieser Service in IAM nicht erkannt wird.
API Gateway	None	In der Richtlinie wird kein Zugriff festgelegt.
CodeBuild	⚠ Es sind keine Aktionen definiert.	Kein Zugriff, da keine Aktionen für den Service definiert sind. Weitere Informationen zum Verständnis und zum Beheben dieses Problems finden Sie unter the section called “Meine Richtlinie erteilt nicht die erwarteten Berechtigungen”.

Wie zuvor erwähnt, gibt Full access (Voller Zugriff) an, dass die Richtlinie Zugriff auf alle Aktionen innerhalb des Services bietet. Richtlinien, die Zugriff auf einige, aber nicht alle Aktionen innerhalb eines Services gewähren, werden entsprechend der Zugriffsebenenklassifizierung gruppiert. Diese Zugriffsebenengruppen sind Folgende:

- Full: Die Richtlinie bietet Zugriff auf alle Aktionen in der angegebenen Zugriffsebenenklassifizierung.
- Limited: Die Richtlinie bietet Zugriff auf eine oder mehrere Aktionen in der angegebenen Zugriffsebenenklassifizierung, jedoch nicht auf alle Aktionen.
- None: Die Richtlinie bietet keinen Zugriff.
- (leer): Dieser Service wird in IAM nicht erkannt. Wenn der Servicename einen Tippfehler enthält, gewährt die Richtlinie keinen Zugriff auf den Service. Wenn der Servicename korrekt ist, werden Richtlinienübersichten von diesem Service möglicherweise nicht unterstützt oder befindet sich in der Vorschau. In diesem Fall kann es sein, dass die Richtlinie Zugriff gewährt, dieser Zugriff kann jedoch nicht in der Richtlinienübersicht angezeigt werden. Informationen dazu, wie Sie

Unterstützung der Richtlinienübersicht für einen allgemein verfügbaren (GA) Service anfordern, finden Sie unter Der Service unterstützt keine IAM-Richtlinienübersichten.

Zusammenfassungen der Zugriffsebenen, die eingeschränkten (teilweisen) Zugriff auf Aktionen beinhalten, werden anhand der AWS Zugriffsebenenklassifizierungen`List`,, `Read` `Tagging``Write`, oder gruppiert. `Permissions management`

AWS Zugriffsebenen

AWS definiert die folgenden Zugriffsebenenklassifizierungen für die Aktionen in einem Dienst:

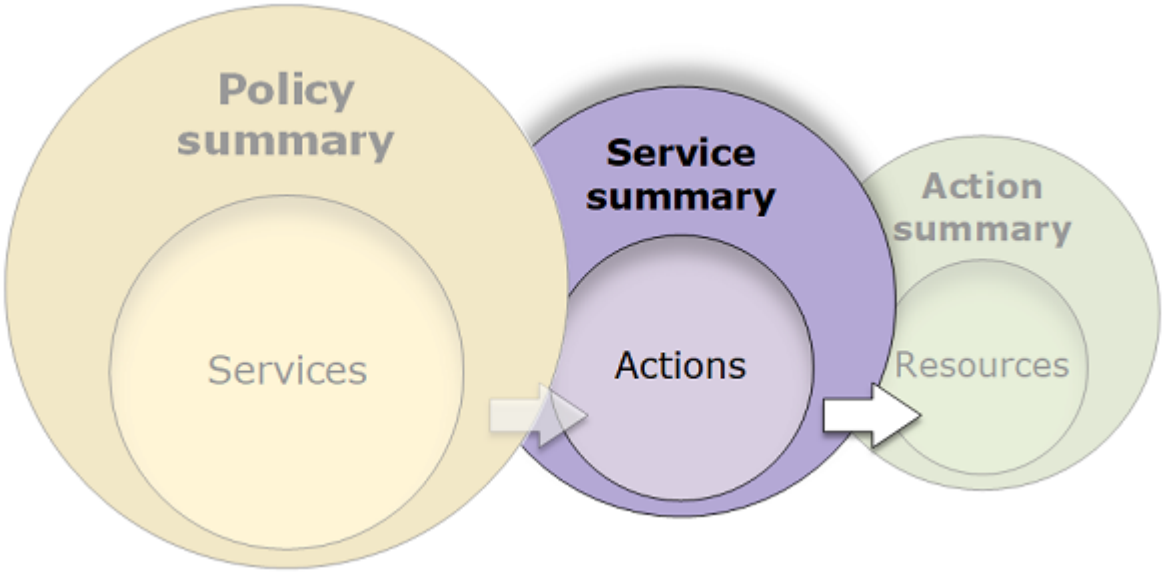
- List: Die Berechtigung zum Auflisten von Ressourcen innerhalb des Services, um zu bestimmen, ob ein Objekt vorhanden ist. Aktionen mit dieser Zugriffsebene können Objekte auflisten, aber nicht die Inhalte einer Ressource sehen. Zum Beispiel hat die Amazon S3-Aktion `ListBucket` die Zugriffsebene List.
- Read: Die Berechtigung zum Lesen, jedoch nicht zum Bearbeiten der Inhalte und Attribute der Ressourcen innerhalb des Services. Zum Beispiel haben die Amazon S3-Aktionen `GetObject` und `GetBucketLocation` die Zugriffsebene Read.
- Tagging: Die Berechtigung zum Ausführen von Aktionen, die nur den Status der Ressourcen-Tags ändern. Beispielsweise verfügen die IAM-Aktionen `TagRole` und `UntagRole` über die Zugriffsebene Tagging (Markieren), da sie nur eine Rolle markieren oder die Markierung der Rolle entfernen können. Die `CreateRole`-Aktion ermöglicht jedoch das Markieren einer Rolle, wenn Sie diese Rolle erstellen. Da die Aktion nicht nur ein Tag hinzufügt, hat sie auch die Zugriffsebene `Write`.
- Write: Die Berechtigung zum Erstellen, Löschen oder Ändern von Ressourcen innerhalb des Services. Beispielsweise verfügen die Amazon S3-Aktionen `CreateBucket`, `DeleteBucket` und `PutObject` über die Zugriffsebene Write. `Write`-Aktionen lassen es möglicherweise auch zu, einen Ressourcen-Tag zu ändern. Jedoch verfügt eine Aktion, die nur Änderungen an Tags zulässt, über die Zugriffsebene `Tagging`.
- Permissions management: Die Berechtigung zum Erteilen oder Ändern von Ressourcenberechtigungen innerhalb des Services. Beispielsweise verfügen die meisten IAM und AWS Organizations Aktionen sowie Aktionen wie die Amazon S3 S3-Aktionen `PutBucketPolicy` `DeleteBucketPolicy` über die Zugriffsebene Permissions Management.

> **Tipp**
>
> Um Ihre Sicherheit zu verbessern AWS-Konto, schränken Sie Richtlinien ein oder überwachen Sie sie regelmäßig, einschließlich der Zugriffsebenenklassifizierung für die Berechtigungsverwaltung.

Informationen zur Klassifizierung der Zugriffsebenen für alle Aktionen in einem Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Serviceübersicht (Liste der Aktionen)

Richtlinien werden in drei Tabellen zusammengefasst: Richtlinienübersicht, Serviceübersicht und Aktionsübersicht. In der Tabelle Serviceübersicht werden die Aktionen und Übersichten der für die Richtlinie des ausgewählten Services definierten Berechtigungen aufgelistet.



Sie können eine Serviceübersicht für jeden aufgeführten Service in der Richtlinienübersicht der Richtlinie anzeigen, die die Berechtigungen erteilt. Die Tabelle ist in die Bereiche Uncategorized actions (Nicht kategorisierte Aktionen) und Uncategorized resource types (Nicht kategorisierte Ressourcentypen) sowie die Zugriffsebene unterteilt. Wenn die Richtlinie eine Aktion enthält, die IAM nicht erkennt, ist die Aktion im Abschnitt Uncategorized actions der Tabelle enthalten. Wenn IAM die Aktion erkennt, ist sie in einem der Zugriffsebenenbereiche (List (Liste), Read (Lesen), Write (Schreiben) und Permissions management (Berechtigungsverwaltung)) der Tabelle enthalten.

Informationen zur Zugriffsebenenklassifizierung, die jeder Aktion in einem Service zugewiesen ist, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Anzeigen von Serviceübersichten

Sie können die Serviceübersicht für alle verwalteten Richtlinien auf der Seite Policies (Richtlinien) einsehen.

So zeigen Sie die Serviceübersicht für eine verwaltete Richtlinie an

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 Policies.
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie anzeigen möchten.
4. Wählen Sie auf der Seite Policy details (Richtliniendetails) der Richtlinie die Registerkarte Permissions (Berechtigungen) aus, um die Richtlinienübersicht anzuzeigen.
5. Wählen Sie in der Serviceliste der Richtlinienübersicht den Namen des anzuzeigenden Service.

So zeigen Sie die Serviceübersicht für eine einem Benutzer zugeordnete Richtlinie an

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 auf Users (Benutzer).
3. Wählen Sie in der Benutzerliste den Namen des Benutzers aus, dessen Richtlinie Sie anzeigen möchten.
4. Wählen Sie auf der Seite Summary (Übersicht) des Benutzers die Registerkarte Permissions (Berechtigungen) aus, um die Liste der Richtlinien anzuzeigen, die direkt oder aus einer Gruppe an den Benutzer angefügt werden.
5. Wählen Sie in der Richtlinientabelle des Benutzers den Namen der anzuzeigenden Richtlinie aus.

   Wenn Sie die Seite Users (Benutzer) geöffnet haben und die Serviceübersicht für eine an diesen Kunden angehängte Richtlinie anzeigen möchten, werden Sie zur Seite Policies (Richtlinien) weitergeleitet. Sie können Serviceübersichten nur auf der Seite Policies (Richtlinien) anzeigen.
6. Wählen Sie Summary (Übersicht) aus. Wählen Sie in der Serviceliste der Richtlinienübersicht den Namen des anzuzeigenden Service.

> **Note**
>
> Wenn es sich bei der ausgewählten Richtlinie um eine Inlinerichtlinie handelt, die direkt an den Benutzer angefügt wird, wird die Serviceübersichtstabelle angezeigt. Wenn die Richtlinie eine Inlinerichtlinie ist, die aus einer Gruppe angefügt wird, wird das JSON-Richtliniendokument für diese Gruppe geöffnet. Wenn es sich um eine verwaltete Richtlinie handelt, wird die Serviceübersicht für diese Richtlinie auf der Seite Policies (Richtlinien) geöffnet.

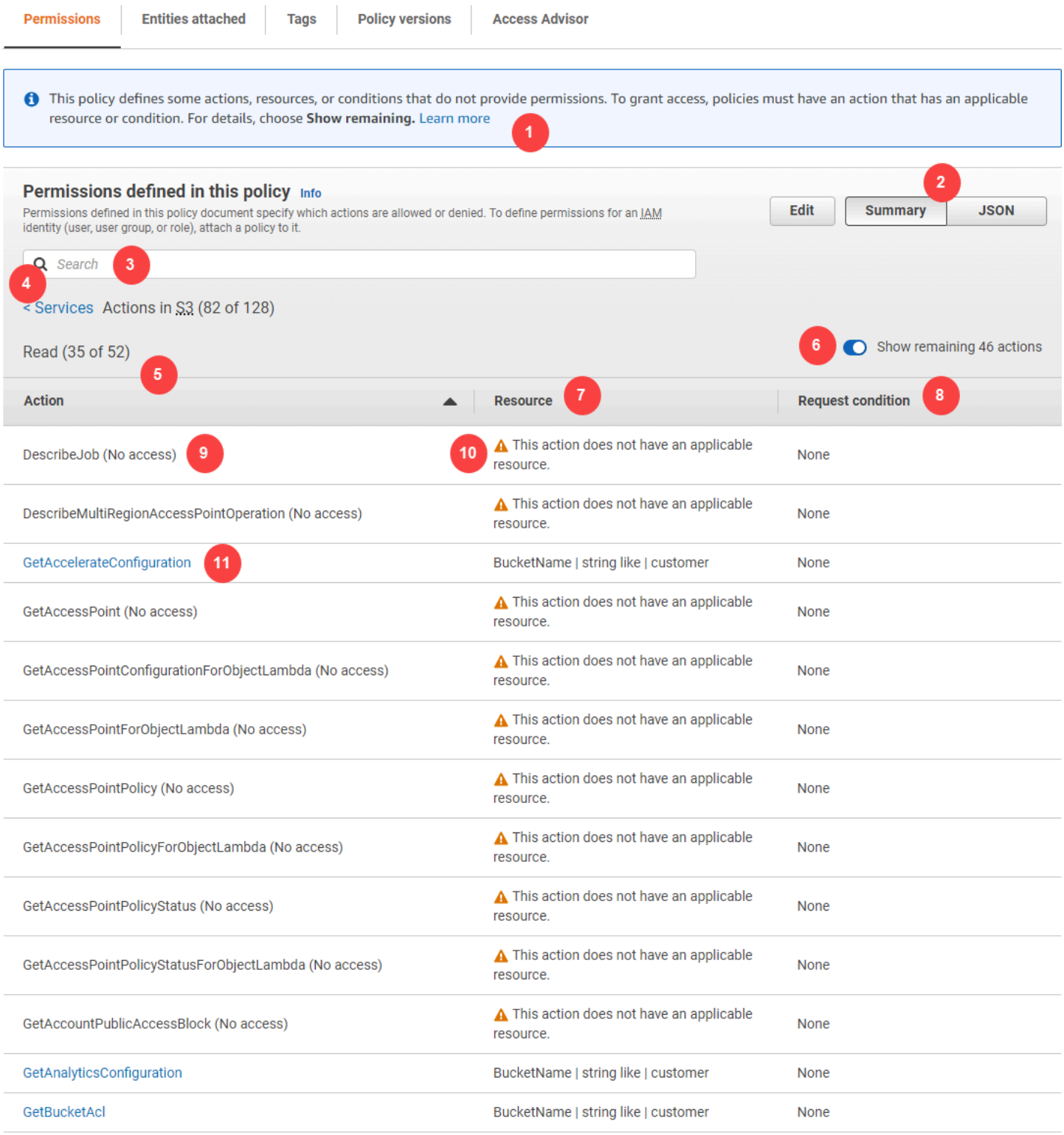
So zeigen Sie die Serviceübersicht für eine einer Rolle zugeordnete Richtlinie an

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 Roles (Rollen).
3. Wählen Sie in der Rollenliste den Namen der Rolle aus, deren Richtlinie Sie anzeigen möchten.
4. Wählen Sie auf der Seite Summary (Zusammenfassung) der Rolle die Registerkarte Permissions (Berechtigungen) aus, um die Liste der Richtlinien anzuzeigen, die der Rolle zugeordnet sind.
5. Wählen Sie in der Richtlinientabelle der Rolle den Namen der anzuzeigenden Richtlinie aus.

   Wenn Sie die Seite Roles (Rollen) geöffnet haben und die Serviceübersicht für eine an diesen Kunden angehängte Richtlinie anzeigen möchten, werden Sie zur Seite Policies (Richtlinien) weitergeleitet. Sie können Serviceübersichten nur auf der Seite Policies (Richtlinien) anzeigen.
6. Wählen Sie in der Serviceliste der Richtlinienübersicht den Namen des anzuzeigenden Service.

Elemente einer Serviceübersicht

Das folgende Beispiel ist die Serviceübersicht für Amazon-S3-Aktionen, die von einer Richtlinienübersicht zugelassen werden. Die Aktionen für diesen Service sind nach Zugriffsebene gruppiert. Von insgesamt 52 Read (Lesen)-Aktionen für den Service sind hier beispielsweise 35 Read (Lesen)-Aktionen definiert.



Die Seite "Serviceübersicht" für verwaltete Richtlinien enthält die folgenden Informationen:

1. Wenn die Richtlinie nicht Berechtigungen für alle Aktionen, Ressourcen und Bedingungen erteilt, die für den Service in der Richtlinie definiert sind, wird oben auf der Seite ein Warnbanner

angezeigt. Die Serviceübersicht enthält dann Details zu dem Problem. Informationen dazu, wie Richtlinienübersichten dabei helfen, Probleme mit den Berechtigungen, die durch Ihre Richtlinie erteilt werden, zu verstehen und zu beheben, finden Sie unter the section called “Meine Richtlinie erteilt nicht die erwarteten Berechtigungen”.

2. Klicken Sie auf JSON, um weitere Informationen zu der Richtlinie anzuzeigen. So können Sie alle Bedingungen für die Aktionen anzeigen. Wenn Sie die Serviceübersicht einer Inlinerichtlinie anzeigen, die direkt einem Benutzer zugeordnet ist, müssen Sie das Dialogfeld "Serviceübersicht" schließen und zur Richtlinienübersicht zurückkehren, um auf das JSON-Richtliniendokument zugreifen zu können.
3. Um die Übersicht für eine bestimmte Aktion anzuzeigen, geben Sie Schlüsselwörter in das Feld Search (Suche) ein, um die Liste der verfügbaren Aktionen zu kürzen.
4. Neben dem Rückwärtspfeil Services finden Sie den Namen des Service (in diesem Fall S3). Die Serviceübersicht für diesen Service umfasst die Liste der in der Richtlinie definierten zulässigen oder verweigerten Aktionen. Wenn der Service auf der Registerkarte Permissions (Berechtigungen) unter (Explicit deny) ((Explizite Ablehnung)) angezeigt wird, werden die in der Serviceübersichtstabelle aufgeführten Aktionen ausdrücklich verweigert. Wenn der Service auf der Registerkarte Permissions (Berechtigungen) unter Allow (Zulassen) angezeigt wird, werden die in der Serviceübersichtstabelle aufgeführten Aktionen zugelassen.
5. Action (Aktion) – Diese Spalte enthält die Aktionen, die in der Richtlinie definiert sind, sowie die Ressourcen und Bedingungen für die einzelnen Aktionen. Wenn die Richtlinie Berechtigungen für die Aktion gewährt verweigert, ist der Aktionsname ein Link zur Tabelle mit der Aktionsübersicht. Die Tabelle gruppiert diese Aktionen in ein bis fünf Bereichen, je nachdem, welche Zugriffsebenen von der Richtlinie gewährt bzw. verweigert werden. Diese Bereiche sind List (Liste), Read (Lesen), Write (Schreiben), Permissions management (Berechtigungsverwaltung) und Tagging (Markieren). Die erste Zahl ist die Anzahl der zulässigen Aktionen, die innerhalb jeder Zugriffsebene Berechtigungen bieten. Die zweite Zahl ist die Gesamtanzahl der Aktionen für den Service. In diesem Beispiel stellen 35 von insgesamt 52 bekannten Read (Lesen)-Aktionen für Amazon S3 Berechtigungen bereit. Informationen zur Zugriffsebenenklassifizierung, die jeder Aktion in einem Service zugewiesen ist, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.
6. Show remaining actions (Die restlichen Aktionen anzeigen) – Klicken Sie auf diese Schaltfläche, um die Tabelle zu erweitern oder zu reduzieren und auch diejenigen Aktionen anzuzeigen, die für diesen Service zwar möglich sind, jedoch keine Berechtigungen bereitstellen. Durch Betätigen der Schaltfläche werden auch Warnungen für Elemente angezeigt, die keine Berechtigungen bereitstellen.

7. Resource (Ressource) – In dieser Spalte werden die Ressourcen aufgelistet, die die Richtlinie für den Service definiert. IAM überprüft nicht, ob die Ressource für jede Aktion gültig ist. In diesem Beispiel sind die Aktionen im Amazon-S3-Service nur für die Amazon-S3-Bucket-Ressource `developer_bucket` zulässig. Je nach den Informationen, die der Service für IAM bereitstellt, wird eventuell ein ARN wie `arn:aws:s3:::developer_bucket/*` oder der definierte Ressourcentyp wie beispielsweise `BucketName = developer_bucket` angezeigt.

   > **Note**
   >
   > Diese Spalte kann eine Ressource aus einem anderen Service enthalten. Wenn die Richtlinienanweisung, die die Ressource umfasst, nicht über die Aktionen und Ressourcen aus demselben Service verfügt, enthält Ihre Richtlinie nicht übereinstimmende Ressourcen. Sie werden nicht von IAM über nicht übereinstimmende Ressourcen gewarnt, wenn Sie eine Richtlinie erstellen oder in der Serviceübersicht anzeigen. In IAM wird auch nicht angegeben, ob die Aktion für die Ressourcen gilt, sondern nur, ob der Service übereinstimmt. Wenn diese Spalte eine nicht übereinstimmende Ressource enthält, sollten Sie Ihre Richtlinien auf Fehler überprüfen. Um Ihre Richtlinien besser zu verstehen, testen Sie sie stets mit dem Richtliniensimulator.

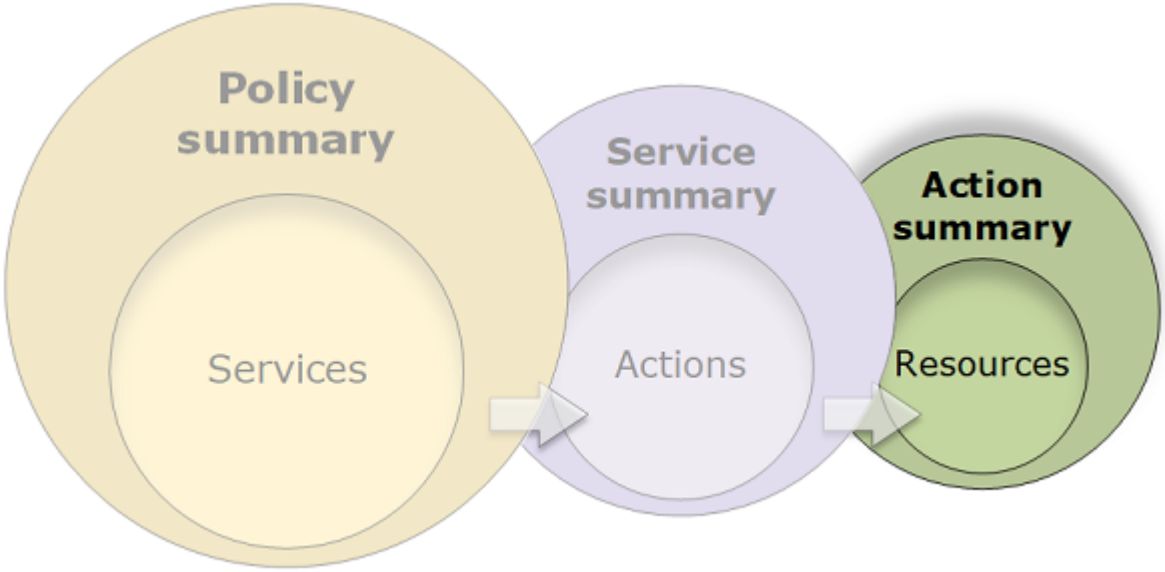
8. Request condition (Anforderungsbedingung) – In dieser Spalte wird angegeben, ob die der Ressource zugeordneten Aktionen Bedingungen unterliegen. Wenn Sie weitere Informationen zu diesen Bedingungen erhalten möchten, wählen Sie JSON aus, um das JSON-Richtliniendokument zu lesen.
9. (No access) (Kein Zugriff) – Diese Richtlinie enthält eine Aktion, die keine Berechtigungen bereitstellt.
10. Resource warning (Ressourcenwarnung) – Bei Aktionen mit Ressourcen, die keine umfassenden Berechtigungen bereitstellen, wird eine der folgenden Warnungen angezeigt:
    - This action does not support resource-level permissions. (Diese Aktion unterstützt keine Berechtigungen auf Ressourcenebene). This requires a wildcard (*) for the resource. (Dies erfordert einen Platzhalter (*) für die Ressource). – Das bedeutet, dass die Richtlinie Berechtigungen auf Ressourcenebene enthält, aber `"Resource": ["*"]` enthalten muss, um Berechtigungen für diese Aktion zu gewähren.
    - This action does not have an applicable resource (Diese Aktion verfügt nicht über eine geeignete Ressource). – Das bedeutet, dass die Aktion ohne eine unterstützte Ressource in der Richtlinien enthalten ist.

- This action does not have an applicable resource and condition (Diese Aktion hat keine anwendbare Ressource und Bedingung). – Das bedeutet, dass die Aktion ohne eine unterstützte Ressource und ohne unterstützte Bedingung in der Richtlinien enthalten ist. In diesem Fall ist auch in der Richtlinie für diesen Service eine Bedingung enthalten. Es gibt jedoch keine Bedingungen, die für diese Aktion gelten.

11 Aktionen, die Berechtigungen bereitstellen, enthalten einen Link zur Aktionsübersicht.

Aktionsübersicht (Liste der Ressourcen)

Richtlinien werden in drei Tabellen zusammengefasst: Richtlinienübersicht, Serviceübersicht und Aktionsübersicht. In der Tabelle Aktionsübersicht finden Sie eine Liste der Ressourcen mit den zugehörigen Bedingungen, die für die ausgewählte Aktion gelten.



Zum Anzeigen einer Aktionsübersicht für einzelne Aktionen, die Berechtigungen erteilen, wählen Sie den Link in der Serviceübersicht. Die Tabelle mit der Aktionsübersicht enthält Informationen zur Ressource sowie Angaben zu ihrer Region und zu ihrem Account (Konto). Sie können auch die Bedingungen anzeigen, die für die jeweilige Ressource gelten. Auf diese Weise können Sie Bedingungen einsehen, die für einige Ressourcen gelten, aber nicht für andere.

Anzeigen von Aktionsübersichten

Die Aktionsübersicht für verwaltete Richtlinien, jede Richtlinie, die an einen Benutzer angehängt ist, und jede Richtlinie, die an eine Rolle angehängt ist, finden Sie auf der Seite Policies (Richtlinien).

So zeigen Sie die Aktionsübersicht für eine verwaltete Richtlinie an

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 Policies.
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie anzeigen möchten.
4. Wählen Sie auf der Seite Policy details (Richtliniendetails) der Richtlinie die Registerkarte Permissions (Berechtigungen) aus, um die Richtlinienübersicht anzuzeigen.
5. Wählen Sie in der Serviceliste der Richtlinienübersicht den Namen des anzuzeigenden Service.
6. Wählen Sie in der Aktionsliste der Serviceübersicht den Namen der anzuzeigenden Aktion.

So zeigen Sie die Aktionsübersicht für eine an einen Benutzer angefügte Richtlinie an

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 Users (Benutzer).
3. Wählen Sie in der Benutzerliste den Namen des Benutzers aus, dessen Richtlinie Sie anzeigen möchten.
4. Wählen Sie auf der Seite Summary (Übersicht) des Benutzers die Registerkarte Permissions (Berechtigungen) aus, um die Liste der Richtlinien anzuzeigen, die direkt oder aus einer Gruppe an den Benutzer angefügt werden.
5. Wählen Sie in der Richtlinientabelle des Benutzers den Namen der anzuzeigenden Richtlinie aus.

   Wenn Sie die Seite Users (Benutzer) geöffnet haben und die Serviceübersicht für eine an diesen Kunden angehängte Richtlinie anzeigen möchten, werden Sie zur Seite Policies (Richtlinien) weitergeleitet. Sie können Serviceübersichten nur auf der Seite Policies (Richtlinien) anzeigen.
6. Wählen Sie in der Serviceliste der Richtlinienübersicht den Namen des anzuzeigenden Service.

> **Note**
>
> Wenn es sich bei der ausgewählten Richtlinie um eine Inlinerichtlinie handelt, die direkt an den Benutzer angefügt wird, wird die Serviceübersichtstabelle angezeigt. Wenn die Richtlinie eine Inlinerichtlinie ist, die aus einer Gruppe angefügt wird, wird das JSON-Richtliniendokument für diese Gruppe geöffnet. Wenn es sich um eine verwaltete

Richtlinie handelt, wird die Serviceübersicht für diese Richtlinie auf der Seite Policies (Richtlinien) geöffnet.

7. Wählen Sie in der Aktionsliste der Serviceübersicht den Namen der anzuzeigenden Aktion.

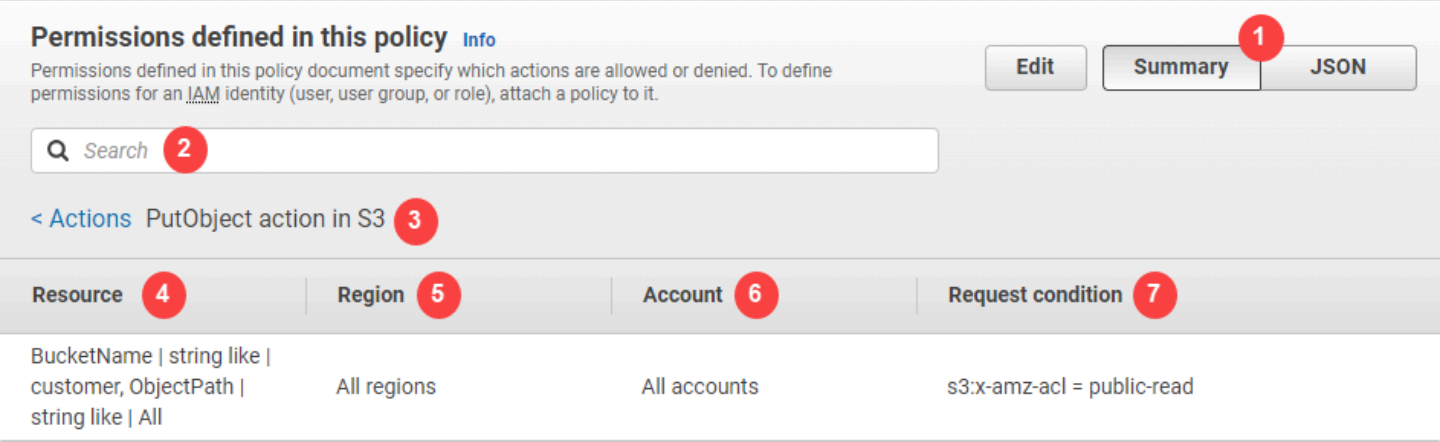
So zeigen Sie die Aktionsübersicht für eine an eine Rolle angefügte Richtlinie an

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 Rollen aus.
3. Wählen Sie in der Rollenliste den Namen der Rolle aus, deren Richtlinie Sie anzeigen möchten.
4. Wählen Sie auf der Seite Summary (Zusammenfassung) der Rolle die Registerkarte Permissions (Berechtigungen) aus, um die Liste der Richtlinien anzuzeigen, die der Rolle zugeordnet sind.
5. Wählen Sie in der Richtlinientabelle der Rolle den Namen der anzuzeigenden Richtlinie aus.

   Wenn Sie die Seite Roles (Rollen) geöffnet haben und die Serviceübersicht für eine an diesen Kunden angehängte Richtlinie anzeigen möchten, werden Sie zur Seite Policies (Richtlinien) weitergeleitet. Sie können Serviceübersichten nur auf der Seite Policies (Richtlinien) anzeigen.
6. Wählen Sie in der Serviceliste der Richtlinienübersicht den Namen des anzuzeigenden Service.
7. Wählen Sie in der Aktionsliste der Serviceübersicht den Namen der anzuzeigenden Aktion.

Elemente einer Aktionsübersicht

Im folgenden Beispiel ist die Aktionsübersicht für die Aktion `PutObject` (Schreiben) aus der Amazon S3-Serviceübersicht angegeben (siehe Serviceübersicht (Liste der Aktionen)). Für diese Aktion definiert die Richtlinie mehrere Bedingungen für eine einzelne Ressource.



Die Aktionsübersichtsseite enthält die folgenden Informationen:

1. Wählen Sie JSON aus, um weitere Details zur Richtlinie anzuzeigen, wie z. B. mehrere Bedingungen, die auf die Aktionen angewendet werden. (Wenn Sie die Serviceübersicht einer Inlinerichtlinie anzeigen, die direkt einem Benutzer zugeordnet ist, müssen Sie das Dialogfeld "Serviceübersicht" schließen und zur Richtlinienübersicht zurückkehren, um auf das JSON-Richtliniendokument zugreifen zu können.)
2. Zum Anzeigen der Übersicht für eine bestimmte Ressource geben Sie Schlüsselwörter in das Feld Search (Suche) ein, um die Liste der verfügbaren Ressourcen zu kürzen.
3. Neben dem Zurück-Pfeil „Aktionen“ werden der Name des Dienstes und der Aktion im Format angezeigt `action name action in service` (in diesem Fall PutObjectAktion in S3). Die Aktionsübersicht für diesen Service umfasst die Liste der in der Richtlinie definierten Ressourcen.
4. Resource – In dieser Spalte werden die Ressourcen aufgelistet, die die Richtlinie für den ausgewählten Service definiert. In diesem Beispiel ist die PutObjectAktion für alle Objektpfade zulässig, jedoch nur für die `developer_bucket` Amazon S3 S3-Bucket-Ressource. Je nach den Informationen, die der Service für IAM bereitstellt, wird eventuell ein ARN wie `arn:aws:s3:::developer_bucket/*` oder der definierte Ressourcentyp wie beispielsweise `BucketName = developer_bucket, ObjectPath = All` angezeigt.
5. Region – In dieser Spalte wird die Region angezeigt, in der die Ressource definiert ist. Ressourcen können für alle Regionen oder eine einzelne Region definiert werden. Sie können nur in einer bestimmten Region vorhanden sein.
   - All regions (Alle Regionen) – Die Aktionen, die der Ressource zugeordnet sind, gelten für alle Regionen. In diesem Beispiel gehört die Aktion zu einem globalen Dienst, Amazon S3. Aktionen, die zu globalen Services gehören, gelten für alle Regionen.

- Region text Die der Ressource zugeordneten Aktionen gelten für eine Region. Eine Richtlinie kann z. B. die Region `us-east-2` für eine Ressource festlegen.

6. Account (Konto) – In dieser Spalte wird angegeben, ob die der Ressource zugeordneten Services oder Aktionen für ein bestimmtes Konto gelten. Ressourcen können in allen Konten oder in einem einzelnen Konto vorhanden sein. Sie können nur in einem bestimmten Konto vorhanden sein.
   - All accounts (Alle Konten) – Die Aktionen, die der Ressource zugeordnet sind, gelten für alle Konten. In diesem Beispiel gehört die Aktion zu einem globalen Dienst, Amazon S3. Aktionen, die zu globalen Services gehören, gelten für alle Konten.
   - This account (Dieses Konto) – Die der Ressource zugeordneten Aktionen gelten nur für das aktuelle Konto.
   - Account number Die der Ressource zugeordneten Aktionen gelten für ein Konto (eins, bei dem Sie zurzeit nicht angemeldet sind). Wenn beispielsweise eine Richtlinie das Konto `123456789012` für eine Ressource festlegt, wird die Kontonummer in der Richtlinienübersicht angezeigt.
7. Request condition – In dieser Spalte wird angegeben, ob die der Ressource zugeordneten Aktionen Bedingungen unterliegen. Dieses Beispiel enthält die Bedingung `s3:x-amz-acl = public-read`. Wenn Sie weitere Informationen zu diesen Bedingungen erhalten möchten, wählen Sie JSON aus, um das JSON-Richtliniendokument zu lesen.

Beispiele für Richtlinienübersichten

Die folgenden Beispiele enthalten JSON-Richtlinien mit ihren zugehörigen Richtlinienübersichten, den Serviceübersichten und den Aktionsübersichten, die Ihnen helfen, die über eine Richtlinie erteilten Berechtigungen zu verstehen.

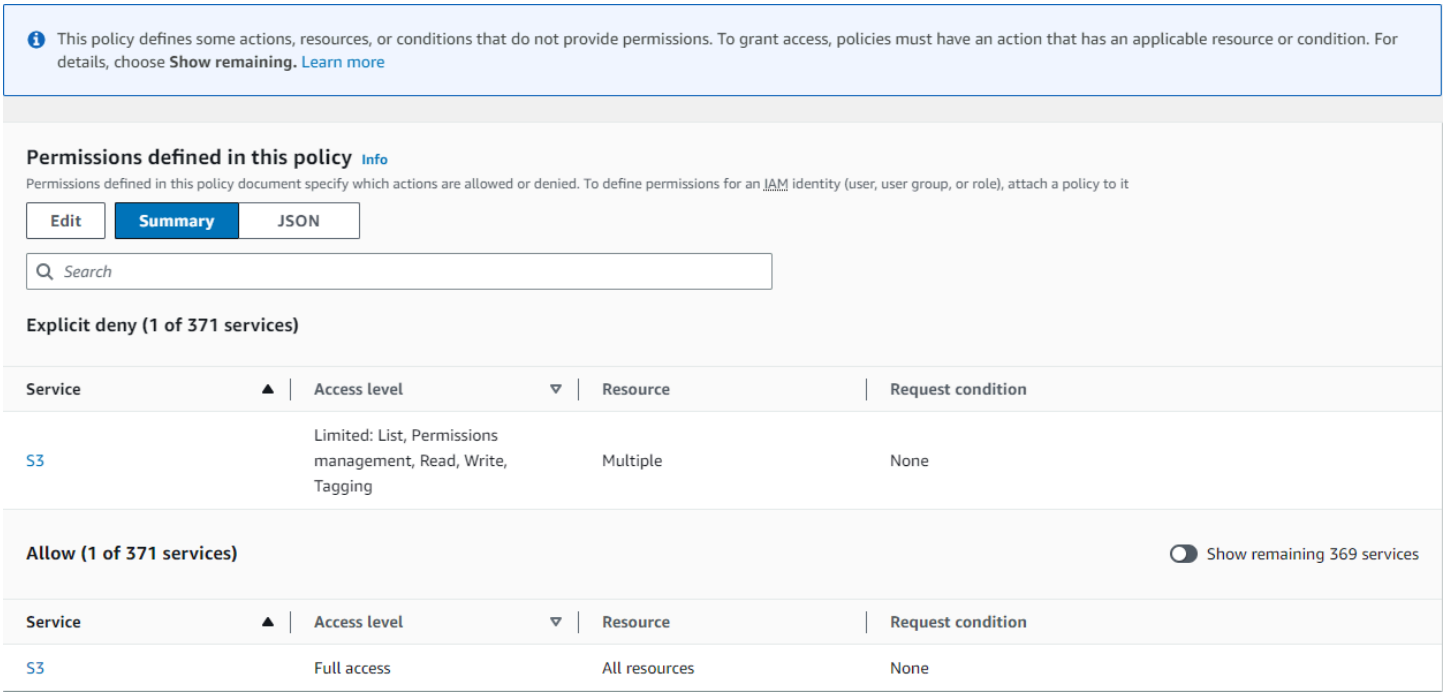
Richtlinie 1: DenyCustomerBucket

Diese Richtlinie enthält eine Zugriffsberechtigung und eine Zugriffsverweigerung für denselben Service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
```

```
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucketZusammenfassung der Richtlinie:



DenyCustomerBucket Zusammenfassung des S3-Dienstes (Explizite Ablehnung):

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject Zusammenfassung der Aktion (gelesen):

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

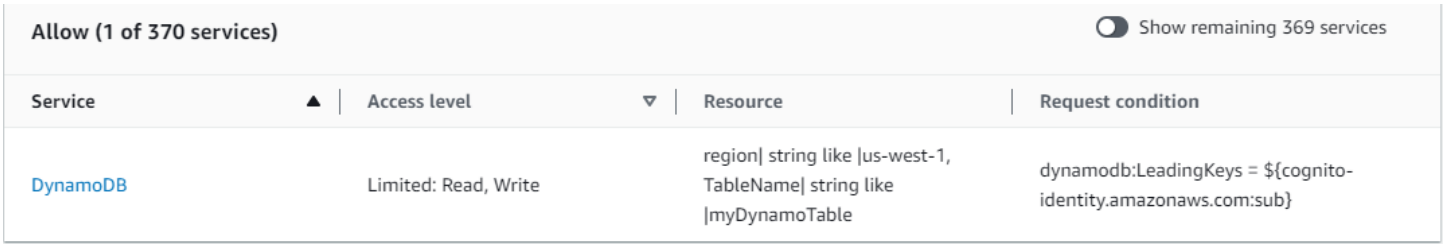
Richtlinie 2: DynamoDbRowCognito ID

Diese Richtlinie ermöglicht den zeilenweisen Zugriff auf Amazon DynamoDB basierend auf der Amazon Cognito-ID des Benutzers.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoZusammenfassung der ID-Richtlinie:



Allow (1 of 370 services) — Show remaining 369 services

Service	Access level	Resource	Request condition
DynamoDB	Limited: Read, Write	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

DynamoDbRowCognitoID DynamoDB (Allow) Serviceübersicht:

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

GetItem Zusammenfassung der Aktionen (Liste):

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

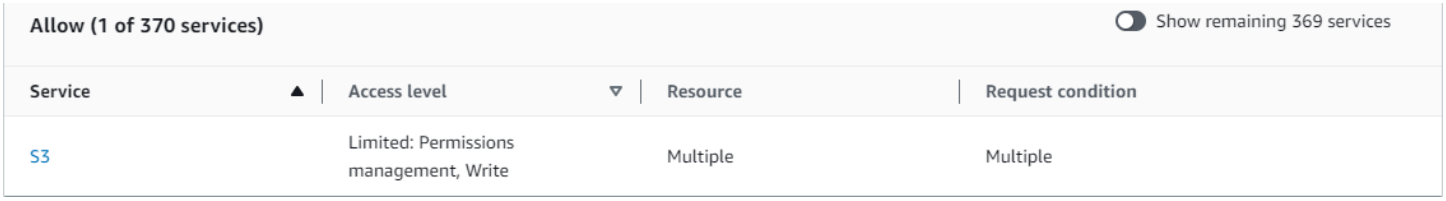
Richtlinie 3: MultipleResourceCondition

Diese Richtlinie umfasst mehrere Ressourcen und Bedingungen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

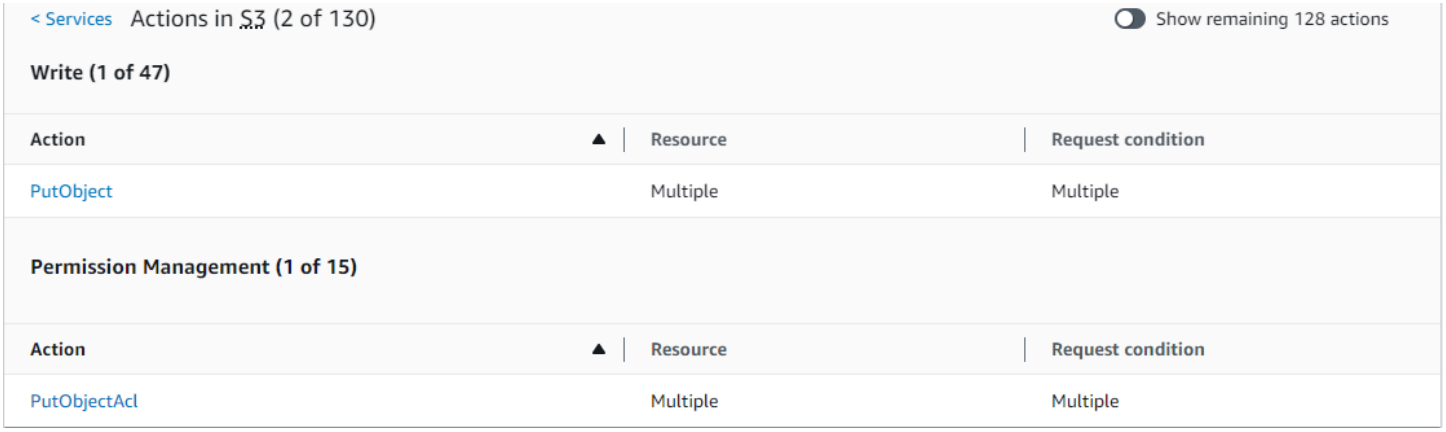
MultipleResourceConditionZusammenfassung der Richtlinie:



Allow (1 of 370 services) — Show remaining 369 services

Service	Access level	Resource	Request condition
S3	Limited: Permissions management, Write	Multiple	Multiple

MultipleResourceCondition Zusammenfassung des S3-Dienstes (Zulassen):



< Services Actions in S3 (2 of 130) — Show remaining 128 actions

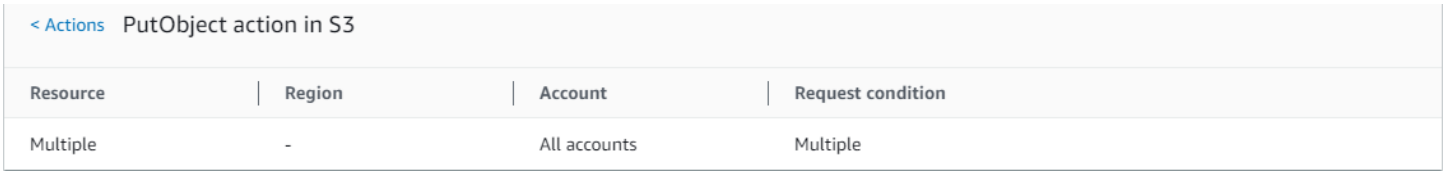
Write (1 of 47)

Action	Resource	Request condition
PutObject	Multiple	Multiple

Permission Management (1 of 15)

Action	Resource	Request condition
PutObjectAcl	Multiple	Multiple

PutObject Zusammenfassung der Aktion (Schreiben):



< Actions PutObject action in S3

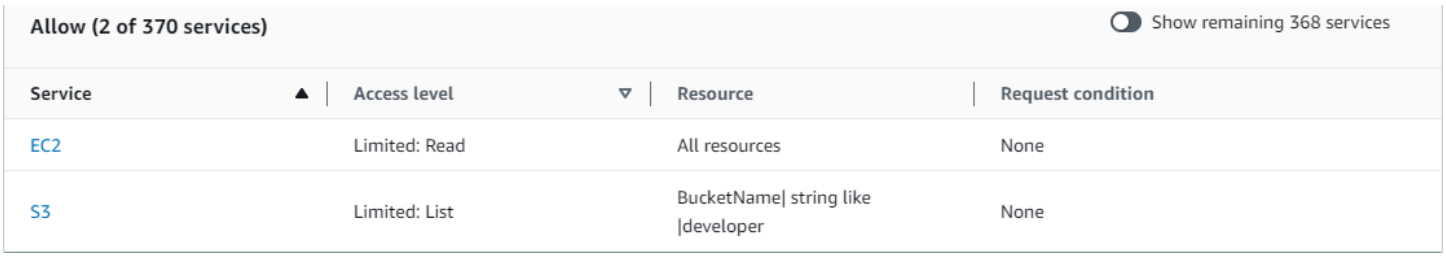
Resource	Region	Account	Request condition
Multiple	-	All accounts	Multiple

Richtlinie 4: EC2_Troubleshoot

Die folgende Richtlinie ermöglicht es den Benutzern, einen Screenshot von einer laufenden Amazon EC2-Instance zu erstellen, um die EC2-Fehlerbehebung zu unterstützen. Diese Richtlinie ermöglicht es außerdem, Informationen zu den Elementen im Amazon S3-Entwickler-Bucket anzuzeigen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Troubleshoot Richtlinienübersicht:



Allow (2 of 370 services) — Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

EC2_Troubleshoot S3 (Allow) Serviceübersicht:

< Services Actions in S3 (1 of 130) Show remaining 129 actions

List (1 of 7)

Action	Resource	Request condition
ListBucket	BucketName| string like |developer	None

ListBucket (Liste) Zusammenfassung der Aktion:

< Actions ListBucket action in S3

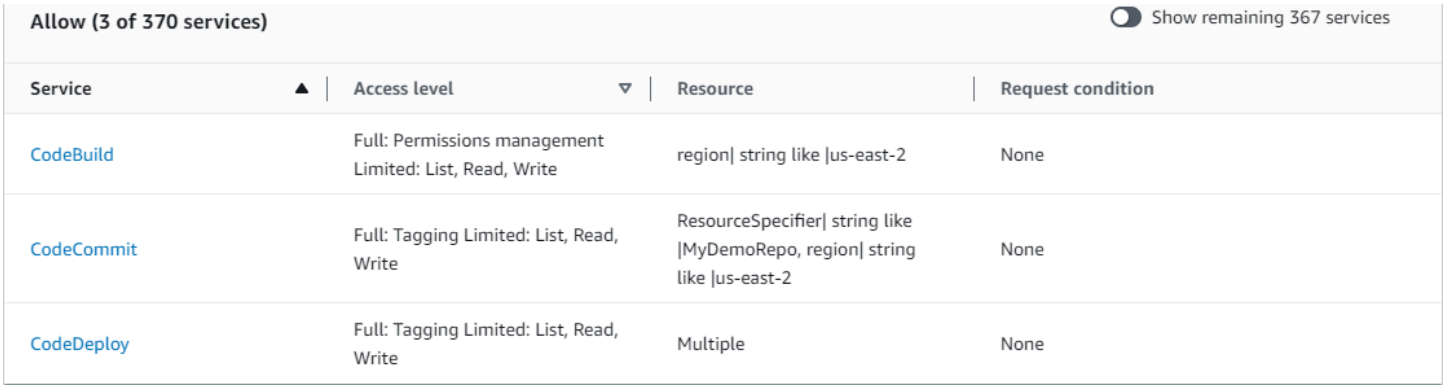
Resource	Region	Account	Request condition
BucketName| string like |developer	-	All accounts	None

Richtlinie 5: CodeBuild _ CodeCommit _ CodeDeploy

Diese Richtlinie bietet Zugriff auf spezifische CodeBuild CodeCommit, und CodeDeploy Ressourcen. Da diese Ressourcen für jeden Service spezifisch sind, erscheinen Sie nur zusammen mit dem zugehörigen Service. Wenn Sie eine Ressource aufführen, die keinem Service im `Action`-Element entspricht, erscheint die Ressource in allen Aktionsübersichten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
            ]
        }
    ]
}
```

CodeBuild_ CodeCommit _ Zusammenfassung CodeDeploy der Richtlinien:



Allow (3 of 370 services) Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

CodeBuild_ CodeCommit _ Serviceübersicht CodeDeploy CodeBuild (zulassen):

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

CodeBuild_ CodeCommit _ Zusammenfassung der Aktion CodeDeploy StartBuild (Schreiben):

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen

Ressourcen sind Objekte innerhalb eines Services. Zu den IAM-Ressourcen gehören Gruppen, Benutzer, Rollen und Richtlinien. Wenn Sie mit Root-Benutzer des AWS-Kontos -Anmeldeinformationen angemeldet sind, gibt es keine Einschränkungen bei der Verwaltung von IAM-Anmeldeinformationen oder IAM-Ressourcen. Allerdings müssen den IAM-Benutzern explizit Berechtigungen für die Verwaltung von Anmeldeinformationen oder IAM-Ressourcen erteilt werden. Sie können dies durch das Zuweisen einer identitätsbasierten Richtlinie zum Benutzer erledigen.

> **Note**
>
> Wenn wir in der gesamten AWS Dokumentation von einer IAM-Richtlinie sprechen, ohne eine der spezifischen Kategorien zu erwähnen, meinen wir damit eine identitätsbasierte, vom Kunden verwaltete Richtlinie. Details zu den Richtlinienkategorien finden Sie unter the section called “Richtlinien und Berechtigungen”.

Berechtigungen für die Verwaltung von IAM-Identitäten

Die Berechtigungen, die für die Verwaltung von IAM-Gruppen, -Benutzern, -Rollen und -Anmeldeinformationen erforderlich sind, entsprechen in der Regel den API-Aktionen für die Aufgabe. Um z. B. IAM-Benutzer anzulegen, benötigen Sie die `iam:CreateUser`-Berechtigung, die der entsprechende API-Befehl hat: `CreateUser`. Um einem IAM-Benutzer zu erlauben, andere IAM-Benutzer zu erstellen, können Sie diesem Benutzer zum Beispiel die folgende IAM-Richtlinie zuweisen:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

In einer Richtlinie ist der Wert des Elements `Resource` von der Aktion abhängig und auf welche Ressourcen diese Aktion Einfluss nimmt. Im vorherigen Beispiel erteilt die Richtlinie dem Benutzer die Berechtigung, alle möglichen Benutzer zu erstellen (* ist ein Platzhalter, der

mit allen Zeichenfolgen übereinstimmt). Im Gegensatz dazu verfügt eine Richtlinie, über die Benutzer nur ihre eigenen Zugriffsschlüssel ändern können (API-Aktionen `CreateAccessKey` und `UpdateAccessKey`), in der Regel über ein `Resource`-Element. In diesem Fall enthält der ARN eine Variable (`${aws:username}`), die sich, wie im folgenden Beispiel, in den Namen des aktuellen Benutzers auflöst:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Im vorherigen Beispiel ist `${aws:username}` eine Variable, die den Benutzernamen auf den aktuellen Benutzer auflöst. Weitere Informationen zu Richtlinienvariablen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Mit einem Platzhalterzeichen (*) im Aktionsnamen können Sie oft das Erteilen von Berechtigungen für alle Aktionen einer bestimmten Aufgabe einfacher gestalten. Wenn Sie beispielsweise Benutzern die Berechtigung zur Durchführung aller IAM-Aktionen erteilen möchten, können Sie `iam:*` für die Aktion verwenden. Um den Benutzern zu erlauben, die Aktionen auszuführen, die sich nur auf die Berechtigung zur Durchführung von Zugriffsschlüsseln beziehen, können Sie `iam:*AccessKey*` im Element `Action` in einer Richtlinienanweisung verwenden. So erhält der Benutzer die Berechtigung zum Ausführen der Aktionen `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`,`ListAccessKeys` und `UpdateAccessKey`. (Wenn in future eine

Aktion zu IAM hinzugefügt wird, deren Name "AccessKey" enthält, erhält der Benutzer durch die Verwendung von `iam:*AccessKey*` for das `Action` Element auch die Berechtigung für diese neue Aktion.) Das folgende Beispiel zeigt eine Richtlinie, die es Benutzern ermöglicht, alle Aktionen auszuführen, die sich auf ihre eigenen Zugriffsschlüssel beziehen (*`account-id`*ersetzen Sie sie durch Ihre AWS-Konto ID):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/${aws:username}"
  }
}
```

Für einige Aufgaben, wie z. B. das Löschen einer Gruppe, sind mehrere Aktionen erforderlich: Sie müssen zunächst die Benutzer aus der Gruppe entfernen, dann die Gruppenrichtlinien trennen oder löschen und schließlich die Gruppe löschen. Wenn Sie einem Benutzer die Berechtigung zum Löschen einer Gruppe erteilen möchten, müssen Sie sicherstellen, dass Sie ihm alle Berechtigung zur Ausführung der damit zusammenhängenden Aktionen erteilen.

Berechtigungen zum Arbeiten in der AWS Management Console

In den vorherigen Beispielen sind Richtlinien aufgeführt, die dem Benutzer die Berechtigung erteilen, Aktionen mit der AWS CLI oder den AWS -SDKs auszuführen.

Während die Benutzer mit der Konsole arbeiten, stellt die Konsole Anfragen an IAM um Gruppen, Benutzer, Rollen und Richtlinien sowie einer Gruppe zugeordnete Richtlinien abzurufen. Die Konsole gibt auch Anfragen zum Abrufen von AWS-Konto Informationen und Informationen über den Principal aus. Der Auftraggeber ist der Benutzer, der in der Konsole Anforderungen stellt.

Im Allgemeinen müssen Sie für die Ausführung einer Aktion nur die passende Aktion in einer Richtlinie haben. Um einen Benutzer anzulegen, benötigen Sie die Berechtigung zum Aufrufen der Aktion `CreateUser`. Wenn Sie die Konsole verwenden, um eine Aktion auszuführen, müssen Sie über die Berechtigung zum Anzeigen, Auflisten, Abrufen oder anderweitigen Anzeigen einer Ressource in der Konsole verfügen. Dies ist notwendig, damit Sie durch die Konsole navigieren können, um die angegebene Aktion auszuführen. Wenn beispielsweise der Benutzer Jorge die Konsole zum Ändern seines eigenen Zugriffsschlüssel verwenden möchte, ruft er die IAM-Konsole auf und klickt auf Users. Diese Aktion bewirkt, dass die Konsole eine `ListUsers`-Anforderung

stellt. Wenn Jorge nicht über die Berechtigung für die Aktion `iam:ListUsers` verfügt, verweigert die Konsole den Zugriff beim Versuch, die Benutzer aufzulisten. Daher kann Jorge nicht seinen eigenen Namen und Zugriffsschlüssel ermitteln, selbst wenn er Berechtigungen für die Aktionen `CreateAccessKey` und `UpdateAccessKey` hat.

Wenn Sie Benutzern Berechtigungen zur Verwaltung von Gruppen, Benutzern, Rollen, Richtlinien und Anmeldeinformationen mit der erteilen möchten AWS Management Console, müssen Sie Berechtigungen für die Aktionen angeben, die die Konsole ausführt. Beispiele für Richtlinien, die Sie verwenden können, um einem Benutzer diese Berechtigungen zu erteilen, finden Sie unter Beispielrichtlinien für die Verwaltung von IAM-Ressourcen.

Gewähren von Berechtigungen über AWS -Konten hinweg

Sie können direkt in Ihrem eigenen Konto IAM-Benutzern Zugriff auf Ihre Ressourcen gewähren. Wenn Benutzer von einem anderen Konto aus Zugriff auf Ihre Ressourcen benötigen, können Sie eine IAM-Rolle erstellen, bei der es sich um eine Entität handelt, die Berechtigungen enthält, jedoch keinem bestimmten Benutzer zugeordnet ist. Benutzer von anderen Konten können dann die Rolle verwenden und entsprechend den Berechtigungen, die Sie der Rolle zugeordnet haben, auf Ressourcen zugreifen. Weitere Informationen finden Sie unter Bereitstellen des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

> **Note**
>
> Einige Services unterstützen ressourcenbasierte Richtlinien wie in Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien beschrieben (wie Amazon S3, Amazon SNS und Amazon SQS). Für diese Services besteht eine Alternative zur Verwendung von Rollen darin, eine Richtlinie an die Ressource (Bucket, Thema oder Warteschlange) anzuhängen, die Sie freigeben möchten. Mit der ressourcenbasierten Richtlinie kann das AWS Konto angegeben werden, das über Berechtigungen für den Zugriff auf die Ressource verfügt.

Service-Berechtigungen für den Zugriff auf einen anderen Service

Viele AWS Dienste greifen auf andere AWS Dienste zu. Beispielsweise verwalten mehrere AWS Dienste — darunter Amazon EMR, Elastic Load Balancing und Amazon EC2 Auto Scaling — Amazon EC2 EC2-Instances. Andere AWS Dienste verwenden Amazon S3 S3-Buckets, Amazon SNS SNS-Themen, Amazon SQS-Warteschlangen usw.

Das Szenario für das Verwalten von Berechtigungen in diesen Fällen variiert je nach Service. Hier finden Sie einige Beispiele für die Art und Weise, wie Berechtigungen für verschiedene Services verarbeitet werden:

- In Amazon EC2 Auto Scaling müssen Benutzer über die Berechtigung verfügen, Auto Scaling zu verwenden. Es muss ihnen jedoch nicht explizit die Berechtigung erteilt werden, Amazon EC2-Instances zu verwalten.
- In legt eine IAM-Rolle fest AWS Data Pipeline, was eine Pipeline tun kann. Benutzer benötigen eine entsprechende Genehmigung, um die Rolle übernehmen zu können. (Detaillierte Informationen finden Sie unter Granting Permissions to Pipelines with IAM im AWS Data Pipeline Developer Guide.)

Einzelheiten zur ordnungsgemäßen Konfiguration von Berechtigungen, sodass ein AWS Dienst die von Ihnen beabsichtigten Aufgaben ausführen kann, finden Sie in der Dokumentation für den Dienst, den Sie aufrufen. Wie Sie eine Rolle für einen Service anlegen, erfahren Sie in Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS -Service.

Konfigurieren eines Service mit einer IAM-Rolle, sodass er in Ihrem Namen arbeitet

Wenn Sie einen AWS Dienst so konfigurieren möchten, dass er in Ihrem Namen funktioniert, geben Sie in der Regel den ARN für eine IAM-Rolle an, die definiert, was der Dienst tun darf. AWS überprüft, ob Sie berechtigt sind, eine Rolle an einen Dienst zu übergeben. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann.

Erforderliche Aktionen

Aktionen stellen die Dinge dar, die Sie mit einer Ressource machen können (z. B. Anzeigen, Anlegen, Bearbeiten und Löschen der Ressource). Aktionen werden von jedem AWS Dienst definiert.

Damit ein jemand eine Aktion ausführen kann, müssen Sie die erforderlichen Aktionen in eine Richtlinie aufnehmen, die für die aufrufende Identität oder die betroffene Ressource gilt. Im Allgemeinen müssen Sie, um die für die Ausführung einer Aktion erforderliche Berechtigung zu erteilen, diese Aktion in Ihre Richtlinie aufnehmen. Um beispielsweise einen Benutzer zu erstellen, müssen Sie die CreateUser Aktion zu Ihrer Richtlinie hinzufügen.

In einigen Fällen kann die eine Aktion erfordern, dass Sie zusätzliche Aktionen in Ihre Richtlinie aufnehmen. Um z. B. jemandem die Berechtigung zu geben, ein Verzeichnis in AWS Directory

Service mit der `ds:CreateDirectory`-Operation zu erstellen, müssen Sie die folgenden Aktionen in ihre Richtlinie aufnehmen:

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:CreateSecurityGroup`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Wenn Sie eine Richtlinie mit dem visuellen Editor erstellen oder bearbeiten, erhalten Sie Warnungen und Aufforderungen, die Sie bei der Auswahl aller erforderlichen Aktionen für Ihre Richtlinie unterstützen.

Weitere Informationen zu den Berechtigungen, die zum Erstellen eines Verzeichnisses erforderlich sind AWS Directory Service, finden Sie unter Beispiel 2: Erlauben Sie einem Benutzer, ein Verzeichnis zu erstellen.

Beispielrichtlinien für die Verwaltung von IAM-Ressourcen

Die nachfolgenden Beispiele enthalten IAM-Richtlinien, mithilfe derer Benutzer Aufgaben wie das Verwalten von IAM-Benutzern, -Gruppen und -Anmeldeinformationen verwalten können. Darunter fallen auch Richtlinien, die es Benutzern ermöglichen, ihre eigenen Passwörter, Zugriffsschlüssel und MFA-Geräte zu verwalten.

Beispiele für Richtlinien, mit denen Benutzer Aufgaben mit anderen AWS Services wie Amazon S3, Amazon EC2 und DynamoDB ausführen können, finden Sie unter. Beispiele für identitätsbasierte Richtlinien in IAM

Themen

- Benutzern erlauben, die Gruppen, Benutzer, Richtlinien und weitere Informationen des Kontos zu Berichtszwecken anzuzeigen
- Benutzern erlauben, Gruppenmitgliedschaften zu verwalten

- Benutzern erlauben, IAM-Benutzer zu verwalten
- Benutzern erlauben, eine Passwortrichtlinie für das Konto festzulegen
- Benutzern erlauben, IAM-Berichte zu Anmeldeinformationen zu erstellen und abzurufen
- Erlauben aller IAM-Aktionen (Administratorzugriff)

Benutzern erlauben, die Gruppen, Benutzer, Richtlinien und weitere Informationen des Kontos zu Berichtszwecken anzuzeigen

Mit der folgenden Richtlinie kann der Benutzer jede IAM-Aktion aufrufen, die mit der Zeichenfolge `Get` oder `List` beginnt, und Berichte erstellen. Informationen zum Anzeigen der Beispielrichtlinie finden Sie unter IAM: Gewährt einen schreibgeschützten Zugriff auf die IAM-Konsole.

Benutzern erlauben, Gruppenmitgliedschaften zu verwalten

Die folgende Richtlinie ermöglicht es dem Benutzer, die Mitgliedschaft der aufgerufenen Gruppe zu aktualisieren. MarketingGroup Informationen zum Anzeigen der Beispielrichtlinie finden Sie unter IAM: Ermöglicht das Verwalten der Mitgliedschaft einer Gruppe sowohl programmgesteuert als auch über die Konsole.

Benutzern erlauben, IAM-Benutzer zu verwalten

Mit der folgenden Richtlinie wird Benutzern ermöglicht, alle Aufgaben auszuführen, die mit der Verwaltung von IAM-Benutzern zusammenhängen. Sie erhalten jedoch keine Berechtigungen für andere Entitäten wie Gruppen oder Richtlinien. Zu den zulässigen Aktionen gehören:

- Erstellen von Benutzern (die Aktion `CreateUser`)
- Löschen von Benutzern. Um diese Aufgabe ausführen zu können, sind alle nachfolgenden Berechtigungen erforderlich: `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup` und `DeleteUser`.
- Auflisten von Benutzern im Konto und in Gruppen (die Aktionen `GetUser`, `ListUsers` und `ListGroupsForUser`).
- Auflisten und Entfernen von Richtlinien eines Benutzers (die Aktionen `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy`, `DeleteUserPolicy`)
- Umbenennen und Ändern des Pfads eines Benutzers (die Aktion `UpdateUser`). Das Element `Resource` muss einen ARN enthalten, der sowohl den Quell- als auch den Zielpfad enthält. Weitere Informationen zu Pfaden finden Sie unter Anzeigenamen und -pfade.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

Mit einigen der Berechtigungen aus der vorhergegangenen Richtlinie wird Benutzern ermöglicht, Aufgaben in der AWS Management Console auszuführen. Benutzer, die benutzerbezogene Aufgaben nur über AWS CLI, die AWS -SDKs oder die IAM-HTTP-Anforderungs-API ausführen,

brauchen möglicherweise nicht alle diese Berechtigungen. Wenn Benutzer beispielsweise bereits den ARN der Richtlinien kennen, deren Zuordnung zu einem Benutzer sie aufheben möchten, benötigen sie die Berechtigung `iam:ListAttachedUserPolicies` nicht. Welche Berechtigungen ein Benutzer genau benötigt, ist abhängig von den Aufgaben, die der Benutzer zur Verwaltung anderer Benutzer ausführen können muss.

Über die folgenden Berechtigungen erhalten Benutzer Zugriff auf Benutzeraufgaben über die AWS Management Console:

- `iam:GetAccount*`
- `iam:ListAccount*`

Benutzern erlauben, eine Passwortrichtlinie für das Konto festzulegen

Sie können ausgewählten Benutzern die Berechtigung zum Abrufen und Aktualisieren der Passwortrichtlinie Ihres AWS-Konto gewähren. Informationen zum Anzeigen der Beispielrichtlinie finden Sie unter IAM: Ermöglicht das Festlegen der Anforderungen an das Kontopasswort programmgesteuert und in der Konsole.

Benutzern erlauben, IAM-Berichte zu Anmeldeinformationen zu erstellen und abzurufen

Sie können Benutzern die Erlaubnis geben, einen Bericht zu erstellen und herunterzuladen, in dem alle Benutzer in Ihrem Bericht aufgeführt sind AWS-Konto. Der Bericht enthält außerdem den Status der verschiedenen Anmeldeinformationen (z. B. Passwörter, Zugriffsschlüssel, MFA-Geräte und Signaturzertifikate). Weitere Informationen zu Berichten zu Anmeldeinformationen finden Sie unter Abrufen von Berichten zu Anmeldeinformationen für Ihr AWS-Konto. Informationen zum Anzeigen der Beispielrichtlinie finden Sie unter IAM: Generieren und Abrufen von -Berichten zu Anmeldeinformationen.

Erlauben aller IAM-Aktionen (Administratorzugriff)

Sie können ausgewählten Benutzern Administratorberechtigungen verliehen, um es ihnen zu ermöglichen, alle IAM-Aktionen einschließlich der Verwaltung von Passwörtern, Zugriffsschlüsseln, MFA-Geräten und Benutzerzertifikaten auszuführen. Mit der folgenden Beispielrichtlinie werden diese Berechtigungen verliehen.

> **Warning**
>
> Wenn Sie einem Benutzer Vollzugriff auf IAM gewähren, kann dieser sich selbst und anderen Benutzern beliebige Berechtigungen verleihen. Der Benutzer kann neue IAM-Entitäten (Benutzer oder Rollen) erstellen und diesen Entitäten Vollzugriff auf alle Ressourcen in Ihrem AWS-Konto gewähren. Wenn Sie einem Benutzer Vollzugriff auf IAM gewähren, erhält dieser somit auch Vollzugriff auf alle Ressourcen in Ihrem AWS-Konto. Das bedeutet, er kann auch alle Ressourcen löschen. Gewähren Sie diese Berechtigungen nur ausgewählten Administratoren, und aktivieren Sie für diese Administratoren Multi-Factor Authentication (MFA).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Codebeispiele für IAM mit SDKs AWS

Die folgenden Codebeispiele zeigen, wie IAM mit einem AWS Software Development Kit (SDK) verwendet wird.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Verwendung DeactivateMfaDevice mit einem AWS SDK oder CLI
- Verwendung DeleteAccessKey mit einem AWS SDK oder CLI
- Verwendung DeleteAccountAlias mit einem AWS SDK oder CLI
- Verwendung DeleteAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteGroup mit einem AWS SDK oder CLI
- Verwendung DeleteGroupPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteInstanceProfile mit einem AWS SDK oder CLI
- Verwendung DeleteLoginProfile mit einem AWS SDK oder CLI
- Verwendung DeleteOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung DeletePolicy mit einem AWS SDK oder CLI
- Verwendung DeletePolicyVersion mit einem AWS SDK oder CLI
- Verwendung DeleteRole mit einem AWS SDK oder CLI
- Verwendung DeleteRolePermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung DeleteRolePolicy mit einem AWS SDK oder CLI
- Verwendung DeleteSAMLProvider mit einem AWS SDK oder CLI
- Verwendung DeleteServerCertificate mit einem AWS SDK oder CLI
- Verwendung DeleteServiceLinkedRole mit einem AWS SDK oder CLI
- Verwendung DeleteSigningCertificate mit einem AWS SDK oder CLI
- Verwendung DeleteUser mit einem AWS SDK oder CLI
- Verwendung DeleteUserPermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung DeleteUserPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteVirtualMfaDevice mit einem AWS SDK oder CLI
- Verwendung DetachGroupPolicy mit einem AWS SDK oder CLI
- Verwendung DetachRolePolicy mit einem AWS SDK oder CLI
- Verwendung DetachUserPolicy mit einem AWS SDK oder CLI
- Verwendung EnableMfaDevice mit einem AWS SDK oder CLI
- Verwendung GenerateCredentialReport mit einem AWS SDK oder CLI
- Verwendung GenerateServiceLastAccessedDetails mit einem AWS SDK oder CLI
- Verwendung GetAccessKeyLastUsed mit einem AWS SDK oder CLI

- Verwendung GetAccountAuthorizationDetails mit einem AWS SDK oder CLI

- Verwendung GetAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung GetAccountSummary mit einem AWS SDK oder CLI
- Verwendung GetContextKeysForCustomPolicy mit einem AWS SDK oder CLI
- Verwendung GetContextKeysForPrincipalPolicy mit einem AWS SDK oder CLI
- Verwendung GetCredentialReport mit einem AWS SDK oder CLI
- Verwendung GetGroup mit einem AWS SDK oder CLI
- Verwendung GetGroupPolicy mit einem AWS SDK oder CLI
- Verwendung GetInstanceProfile mit einem AWS SDK oder CLI
- Verwendung GetLoginProfile mit einem AWS SDK oder CLI
- Verwendung GetOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung GetPolicy mit einem AWS SDK oder CLI
- Verwendung GetPolicyVersion mit einem AWS SDK oder CLI
- Verwendung GetRole mit einem AWS SDK oder CLI
- Verwendung GetRolePolicy mit einem AWS SDK oder CLI
- Verwendung GetSamlProvider mit einem AWS SDK oder CLI
- Verwendung GetServerCertificate mit einem AWS SDK oder CLI
- Verwendung GetServiceLastAccessedDetails mit einem AWS SDK oder CLI
- Verwendung GetServiceLastAccessedDetailsWithEntities mit einem AWS SDK oder CLI
- Verwendung GetServiceLinkedRoleDeletionStatus mit einem AWS SDK oder CLI
- Verwendung GetUser mit einem AWS SDK oder CLI
- Verwendung GetUserPolicy mit einem AWS SDK oder CLI
- Verwendung ListAccessKeys mit einem AWS SDK oder CLI
- Verwendung ListAccountAliases mit einem AWS SDK oder CLI
- Verwendung ListAttachedGroupPolicies mit einem AWS SDK oder CLI
- Verwendung ListAttachedRolePolicies mit einem AWS SDK oder CLI
- Verwendung ListAttachedUserPolicies mit einem AWS SDK oder CLI
- Verwendung ListEntitiesForPolicy mit einem AWS SDK oder CLI
- Verwendung ListGroupPolicies mit einem AWS SDK oder CLI
- Verwendung ListGroups mit einem AWS SDK oder CLI
- Verwendung ListGroupsForUser mit einem AWS SDK oder CLI

- Verwendung ListInstanceProfiles mit einem AWS SDK oder CLI
- Verwendung ListInstanceProfilesForRole mit einem AWS SDK oder CLI
- Verwendung ListMfaDevices mit einem AWS SDK oder CLI
- Verwendung ListOpenIdConnectProviders mit einem AWS SDK oder CLI
- Verwendung ListPolicies mit einem AWS SDK oder CLI
- Verwendung ListPolicyVersions mit einem AWS SDK oder CLI
- Verwendung ListRolePolicies mit einem AWS SDK oder CLI
- Verwendung ListRoleTags mit einem AWS SDK oder CLI
- Verwendung ListRoles mit einem AWS SDK oder CLI
- Verwendung ListSAMLProviders mit einem AWS SDK oder CLI
- Verwendung ListServerCertificates mit einem AWS SDK oder CLI
- Verwendung ListSigningCertificates mit einem AWS SDK oder CLI
- Verwendung ListUserPolicies mit einem AWS SDK oder CLI
- Verwendung ListUserTags mit einem AWS SDK oder CLI
- Verwendung ListUsers mit einem AWS SDK oder CLI
- Verwendung ListVirtualMfaDevices mit einem AWS SDK oder CLI
- Verwendung PutGroupPolicy mit einem AWS SDK oder CLI
- Verwendung PutRolePermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung PutRolePolicy mit einem AWS SDK oder CLI
- Verwendung PutUserPermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung PutUserPolicy mit einem AWS SDK oder CLI
- Verwendung RemoveClientIdFromOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung RemoveRoleFromInstanceProfile mit einem AWS SDK oder CLI
- Verwendung RemoveUserFromGroup mit einem AWS SDK oder CLI
- Verwendung ResyncMfaDevice mit einem AWS SDK oder CLI
- Verwendung SetDefaultPolicyVersion mit einem AWS SDK oder CLI
- Verwendung TagRole mit einem AWS SDK oder CLI
- Verwendung TagUser mit einem AWS SDK oder CLI
- Verwendung UntagRole mit einem AWS SDK oder CLI
- Verwendung UntagUser mit einem AWS SDK oder CLI

- Verwendung UpdateAccessKey mit einem AWS SDK oder CLI
- Verwendung UpdateAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung UpdateAssumeRolePolicy mit einem AWS SDK oder CLI
- Verwendung UpdateGroup mit einem AWS SDK oder CLI
- Verwendung UpdateLoginProfile mit einem AWS SDK oder CLI
- Verwendung UpdateOpenIdConnectProviderThumbprint mit einem AWS SDK oder CLI
- Verwendung UpdateRole mit einem AWS SDK oder CLI
- Verwendung UpdateRoleDescription mit einem AWS SDK oder CLI
- Verwendung UpdateSamlProvider mit einem AWS SDK oder CLI
- Verwendung UpdateServerCertificate mit einem AWS SDK oder CLI
- Verwendung UpdateSigningCertificate mit einem AWS SDK oder CLI
- Verwendung UpdateUser mit einem AWS SDK oder CLI
- Verwendung UploadServerCertificate mit einem AWS SDK oder CLI
- Verwendung UploadSigningCertificate mit einem AWS SDK oder CLI

- Szenarien für IAM mit SDKs AWS
  - Erstellen und verwalten Sie einen ausfallsicheren Service mithilfe eines AWS SDK
  - Erstellen Sie eine IAM-Gruppe und fügen Sie der Gruppe mithilfe eines SDK einen AWS Benutzer hinzu
  - Erstellen Sie einen IAM-Benutzer und übernehmen Sie eine Rolle bei der AWS STS Verwendung eines SDK AWS
  - Erstellen Sie IAM-Benutzer mit Schreibzugriff und Lese-/Schreibzugriff mithilfe eines SDK AWS
  - Verwaltung von IAM-Zugriffsschlüsseln mithilfe eines SDK AWS
  - Verwaltung von IAM-Richtlinien mithilfe eines SDK AWS
  - Verwaltung von IAM-Rollen mithilfe eines SDK AWS
  - Verwalten Sie Ihr IAM-Konto mit einem SDK AWS
  - Rollback einer IAM-Richtlinienversion mithilfe eines SDK AWS
  - Arbeiten Sie mit der IAM Policy Builder-API mithilfe eines SDK AWS

- Codebeispiele für die AWS STS Verwendung von AWS SDKs
  - Aktionen für die AWS STS Verwendung von AWS SDKs
    - Verwendung AssumeRole mit einem AWS SDK oder CLI

- Verwendung AssumeRoleWithWebIdentity mit einem AWS SDK oder CLI
- Verwendung DecodeAuthorizationMessage mit einem AWS SDK oder CLI
- Verwendung GetFederationToken mit einem AWS SDK oder CLI
- Verwendung GetSessionToken mit einem AWS SDK oder CLI

- Szenarien für die AWS STS Verwendung von AWS SDKs
  - Gehen Sie von einer IAM-Rolle aus, für die ein MFA-Token erforderlich ist, und AWS STS verwenden Sie ein SDK AWS
  - Konstruieren Sie mithilfe eines SDK eine AWS URL AWS STS für Verbundbenutzer
  - Rufen Sie AWS STS mithilfe eines SDK ein Sitzungstoken ab, für das ein MFA-Token erforderlich ist AWS

Codebeispiele für IAM mit SDKs AWS

Die folgenden Codebeispiele zeigen, wie IAM mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie eine bestimmte Aufgabe ausführen können, indem Sie mehrere Funktionen innerhalb desselben Services aufrufen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte

Hallo IAM

Die folgenden Codebeispiele veranschaulichen, wie Sie mit der Verwendung von IAM beginnen.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Code für die C MakeLists .txt-CMake-Datei.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code für die iam.cpp-Quellendatei.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Einzelheiten zur API finden Sie unter ListPolicies AWS SDK for C++API-Referenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
```

```
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(context.TODO(), &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
import java.util.List;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      page.Policies.forEach((p) => {
        console.log(`${p.PolicyName}`);
        policyCount++;
      });
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for JavaScript API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Von src/bin/hello.rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), SdkError<ListPoliciesError>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

Von src/iam-service-lib.rs.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;
```

```
    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der API-Referenz zum AWS SDK für Rust.

Codebeispiele

- Aktionen für IAM mithilfe AWS von SDKs
  - Verwendung AddClientIdToOpenIdConnectProvider mit einem AWS SDK oder CLI
  - Verwendung AddRoleToInstanceProfile mit einem AWS SDK oder CLI
  - Verwendung AddUserToGroup mit einem AWS SDK oder CLI
  - Verwendung AttachGroupPolicy mit einem AWS SDK oder CLI
  - Verwendung AttachRolePolicy mit einem AWS SDK oder CLI
  - Verwendung AttachUserPolicy mit einem AWS SDK oder CLI
  - Verwendung ChangePassword mit einem AWS SDK oder CLI
  - Verwendung CreateAccessKey mit einem AWS SDK oder CLI
  - Verwendung CreateAccountAlias mit einem AWS SDK oder CLI
  - Verwendung CreateGroup mit einem AWS SDK oder CLI
  - Verwendung CreateInstanceProfile mit einem AWS SDK oder CLI
  - Verwendung CreateLoginProfile mit einem AWS SDK oder CLI
  - Verwendung CreateOpenIdConnectProvider mit einem AWS SDK oder CLI
  - Verwendung CreatePolicy mit einem AWS SDK oder CLI
  - Verwendung CreatePolicyVersion mit einem AWS SDK oder CLI
  - Verwendung CreateRole mit einem AWS SDK oder CLI

- Verwendung CreateSAMLProvider mit einem AWS SDK oder CLI
- Verwendung CreateServiceLinkedRole mit einem AWS SDK oder CLI
- Verwendung CreateUser mit einem AWS SDK oder CLI
- Verwendung CreateVirtualMfaDevice mit einem AWS SDK oder CLI
- Verwendung DeactivateMfaDevice mit einem AWS SDK oder CLI
- Verwendung DeleteAccessKey mit einem AWS SDK oder CLI
- Verwendung DeleteAccountAlias mit einem AWS SDK oder CLI
- Verwendung DeleteAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteGroup mit einem AWS SDK oder CLI
- Verwendung DeleteGroupPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteInstanceProfile mit einem AWS SDK oder CLI
- Verwendung DeleteLoginProfile mit einem AWS SDK oder CLI
- Verwendung DeleteOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung DeletePolicy mit einem AWS SDK oder CLI
- Verwendung DeletePolicyVersion mit einem AWS SDK oder CLI
- Verwendung DeleteRole mit einem AWS SDK oder CLI
- Verwendung DeleteRolePermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung DeleteRolePolicy mit einem AWS SDK oder CLI
- Verwendung DeleteSAMLProvider mit einem AWS SDK oder CLI
- Verwendung DeleteServerCertificate mit einem AWS SDK oder CLI
- Verwendung DeleteServiceLinkedRole mit einem AWS SDK oder CLI
- Verwendung DeleteSigningCertificate mit einem AWS SDK oder CLI
- Verwendung DeleteUser mit einem AWS SDK oder CLI
- Verwendung DeleteUserPermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung DeleteUserPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteVirtualMfaDevice mit einem AWS SDK oder CLI
- Verwendung DetachGroupPolicy mit einem AWS SDK oder CLI
- Verwendung DetachRolePolicy mit einem AWS SDK oder CLI
- Verwendung DetachUserPolicy mit einem AWS SDK oder CLI
- Verwendung EnableMfaDevice mit einem AWS SDK oder CLI

- Verwendung GenerateCredentialReport mit einem AWS SDK oder CLI
- Verwendung GenerateServiceLastAccessedDetails mit einem AWS SDK oder CLI
- Verwendung GetAccessKeyLastUsed mit einem AWS SDK oder CLI
- Verwendung GetAccountAuthorizationDetails mit einem AWS SDK oder CLI
- Verwendung GetAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung GetAccountSummary mit einem AWS SDK oder CLI
- Verwendung GetContextKeysForCustomPolicy mit einem AWS SDK oder CLI
- Verwendung GetContextKeysForPrincipalPolicy mit einem AWS SDK oder CLI
- Verwendung GetCredentialReport mit einem AWS SDK oder CLI
- Verwendung GetGroup mit einem AWS SDK oder CLI
- Verwendung GetGroupPolicy mit einem AWS SDK oder CLI
- Verwendung GetInstanceProfile mit einem AWS SDK oder CLI
- Verwendung GetLoginProfile mit einem AWS SDK oder CLI
- Verwendung GetOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung GetPolicy mit einem AWS SDK oder CLI
- Verwendung GetPolicyVersion mit einem AWS SDK oder CLI
- Verwendung GetRole mit einem AWS SDK oder CLI
- Verwendung GetRolePolicy mit einem AWS SDK oder CLI
- Verwendung GetSamlProvider mit einem AWS SDK oder CLI
- Verwendung GetServerCertificate mit einem AWS SDK oder CLI
- Verwendung GetServiceLastAccessedDetails mit einem AWS SDK oder CLI
- Verwendung GetServiceLastAccessedDetailsWithEntities mit einem AWS SDK oder CLI
- Verwendung GetServiceLinkedRoleDeletionStatus mit einem AWS SDK oder CLI
- Verwendung GetUser mit einem AWS SDK oder CLI
- Verwendung GetUserPolicy mit einem AWS SDK oder CLI
- Verwendung ListAccessKeys mit einem AWS SDK oder CLI
- Verwendung ListAccountAliases mit einem AWS SDK oder CLI
- Verwendung ListAttachedGroupPolicies mit einem AWS SDK oder CLI
- Verwendung ListAttachedRolePolicies mit einem AWS SDK oder CLI
- Verwendung ListAttachedUserPolicies mit einem AWS SDK oder CLI

- Verwendung ListEntitiesForPolicy mit einem AWS SDK oder CLI
- Verwendung ListGroupPolicies mit einem AWS SDK oder CLI
- Verwendung ListGroups mit einem AWS SDK oder CLI
- Verwendung ListGroupsForUser mit einem AWS SDK oder CLI
- Verwendung ListInstanceProfiles mit einem AWS SDK oder CLI
- Verwendung ListInstanceProfilesForRole mit einem AWS SDK oder CLI
- Verwendung ListMfaDevices mit einem AWS SDK oder CLI
- Verwendung ListOpenIdConnectProviders mit einem AWS SDK oder CLI
- Verwendung ListPolicies mit einem AWS SDK oder CLI
- Verwendung ListPolicyVersions mit einem AWS SDK oder CLI
- Verwendung ListRolePolicies mit einem AWS SDK oder CLI
- Verwendung ListRoleTags mit einem AWS SDK oder CLI
- Verwendung ListRoles mit einem AWS SDK oder CLI
- Verwendung ListSAMLProviders mit einem AWS SDK oder CLI
- Verwendung ListServerCertificates mit einem AWS SDK oder CLI
- Verwendung ListSigningCertificates mit einem AWS SDK oder CLI
- Verwendung ListUserPolicies mit einem AWS SDK oder CLI
- Verwendung ListUserTags mit einem AWS SDK oder CLI
- Verwendung ListUsers mit einem AWS SDK oder CLI
- Verwendung ListVirtualMfaDevices mit einem AWS SDK oder CLI
- Verwendung PutGroupPolicy mit einem AWS SDK oder CLI
- Verwendung PutRolePermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung PutRolePolicy mit einem AWS SDK oder CLI
- Verwendung PutUserPermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung PutUserPolicy mit einem AWS SDK oder CLI
- Verwendung RemoveClientIdFromOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung RemoveRoleFromInstanceProfile mit einem AWS SDK oder CLI
- Verwendung RemoveUserFromGroup mit einem AWS SDK oder CLI
- Verwendung ResyncMfaDevice mit einem AWS SDK oder CLI

- Verwendung SetDefaultPolicyVersion mit einem AWS SDK oder CLI

- Verwendung TagRole mit einem AWS SDK oder CLI
- Verwendung TagUser mit einem AWS SDK oder CLI
- Verwendung UntagRole mit einem AWS SDK oder CLI
- Verwendung UntagUser mit einem AWS SDK oder CLI
- Verwendung UpdateAccessKey mit einem AWS SDK oder CLI
- Verwendung UpdateAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung UpdateAssumeRolePolicy mit einem AWS SDK oder CLI
- Verwendung UpdateGroup mit einem AWS SDK oder CLI
- Verwendung UpdateLoginProfile mit einem AWS SDK oder CLI
- Verwendung UpdateOpenIdConnectProviderThumbprint mit einem AWS SDK oder CLI
- Verwendung UpdateRole mit einem AWS SDK oder CLI
- Verwendung UpdateRoleDescription mit einem AWS SDK oder CLI
- Verwendung UpdateSamlProvider mit einem AWS SDK oder CLI
- Verwendung UpdateServerCertificate mit einem AWS SDK oder CLI
- Verwendung UpdateSigningCertificate mit einem AWS SDK oder CLI
- Verwendung UpdateUser mit einem AWS SDK oder CLI
- Verwendung UploadServerCertificate mit einem AWS SDK oder CLI
- Verwendung UploadSigningCertificate mit einem AWS SDK oder CLI

- Szenarien für IAM mit SDKs AWS
  - Erstellen und verwalten Sie einen ausfallsicheren Service mithilfe eines AWS SDK
  - Erstellen Sie eine IAM-Gruppe und fügen Sie der Gruppe mithilfe eines SDK einen AWS Benutzer hinzu
  - Erstellen Sie einen IAM-Benutzer und übernehmen Sie eine Rolle bei der AWS STS Verwendung eines SDK AWS
  - Erstellen Sie IAM-Benutzer mit Schreibzugriff und Lese-/Schreibzugriff mithilfe eines SDK AWS
  - Verwaltung von IAM-Zugriffsschlüsseln mithilfe eines SDK AWS
  - Verwaltung von IAM-Richtlinien mithilfe eines SDK AWS
  - Verwaltung von IAM-Rollen mithilfe eines SDK AWS
  - Verwalten Sie Ihr IAM-Konto mit einem SDK AWS
  - Rollback einer IAM-Richtlinienversion mithilfe eines SDK AWS
  - Arbeiten Sie mit der IAM Policy Builder-API mithilfe eines SDK AWS

Aktionen für IAM mithilfe AWS von SDKs

Die folgenden Codebeispiele zeigen, wie einzelne IAM-Aktionen mit SDKs ausgeführt werden. AWS Diese Auszüge rufen die IAM-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie unter AWS Identity and Access Management (IAM)-API-Referenz.

Beispiele

- Verwendung AddClientIdToOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung AddRoleToInstanceProfile mit einem AWS SDK oder CLI
- Verwendung AddUserToGroup mit einem AWS SDK oder CLI
- Verwendung AttachGroupPolicy mit einem AWS SDK oder CLI
- Verwendung AttachRolePolicy mit einem AWS SDK oder CLI
- Verwendung AttachUserPolicy mit einem AWS SDK oder CLI
- Verwendung ChangePassword mit einem AWS SDK oder CLI
- Verwendung CreateAccessKey mit einem AWS SDK oder CLI
- Verwendung CreateAccountAlias mit einem AWS SDK oder CLI
- Verwendung CreateGroup mit einem AWS SDK oder CLI
- Verwendung CreateInstanceProfile mit einem AWS SDK oder CLI
- Verwendung CreateLoginProfile mit einem AWS SDK oder CLI
- Verwendung CreateOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung CreatePolicy mit einem AWS SDK oder CLI
- Verwendung CreatePolicyVersion mit einem AWS SDK oder CLI
- Verwendung CreateRole mit einem AWS SDK oder CLI
- Verwendung CreateSAMLProvider mit einem AWS SDK oder CLI
- Verwendung CreateServiceLinkedRole mit einem AWS SDK oder CLI
- Verwendung CreateUser mit einem AWS SDK oder CLI
- Verwendung CreateVirtualMfaDevice mit einem AWS SDK oder CLI
- Verwendung DeactivateMfaDevice mit einem AWS SDK oder CLI

- Verwendung DeleteAccessKey mit einem AWS SDK oder CLI
- Verwendung DeleteAccountAlias mit einem AWS SDK oder CLI
- Verwendung DeleteAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteGroup mit einem AWS SDK oder CLI
- Verwendung DeleteGroupPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteInstanceProfile mit einem AWS SDK oder CLI
- Verwendung DeleteLoginProfile mit einem AWS SDK oder CLI
- Verwendung DeleteOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung DeletePolicy mit einem AWS SDK oder CLI
- Verwendung DeletePolicyVersion mit einem AWS SDK oder CLI
- Verwendung DeleteRole mit einem AWS SDK oder CLI
- Verwendung DeleteRolePermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung DeleteRolePolicy mit einem AWS SDK oder CLI
- Verwendung DeleteSAMLProvider mit einem AWS SDK oder CLI
- Verwendung DeleteServerCertificate mit einem AWS SDK oder CLI
- Verwendung DeleteServiceLinkedRole mit einem AWS SDK oder CLI
- Verwendung DeleteSigningCertificate mit einem AWS SDK oder CLI
- Verwendung DeleteUser mit einem AWS SDK oder CLI
- Verwendung DeleteUserPermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung DeleteUserPolicy mit einem AWS SDK oder CLI
- Verwendung DeleteVirtualMfaDevice mit einem AWS SDK oder CLI
- Verwendung DetachGroupPolicy mit einem AWS SDK oder CLI
- Verwendung DetachRolePolicy mit einem AWS SDK oder CLI
- Verwendung DetachUserPolicy mit einem AWS SDK oder CLI
- Verwendung EnableMfaDevice mit einem AWS SDK oder CLI
- Verwendung GenerateCredentialReport mit einem AWS SDK oder CLI
- Verwendung GenerateServiceLastAccessedDetails mit einem AWS SDK oder CLI
- Verwendung GetAccessKeyLastUsed mit einem AWS SDK oder CLI
- Verwendung GetAccountAuthorizationDetails mit einem AWS SDK oder CLI

- Verwendung GetAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung GetAccountSummary mit einem AWS SDK oder CLI
- Verwendung GetContextKeysForCustomPolicy mit einem AWS SDK oder CLI
- Verwendung GetContextKeysForPrincipalPolicy mit einem AWS SDK oder CLI
- Verwendung GetCredentialReport mit einem AWS SDK oder CLI
- Verwendung GetGroup mit einem AWS SDK oder CLI
- Verwendung GetGroupPolicy mit einem AWS SDK oder CLI
- Verwendung GetInstanceProfile mit einem AWS SDK oder CLI
- Verwendung GetLoginProfile mit einem AWS SDK oder CLI
- Verwendung GetOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung GetPolicy mit einem AWS SDK oder CLI
- Verwendung GetPolicyVersion mit einem AWS SDK oder CLI
- Verwendung GetRole mit einem AWS SDK oder CLI
- Verwendung GetRolePolicy mit einem AWS SDK oder CLI
- Verwendung GetSamlProvider mit einem AWS SDK oder CLI
- Verwendung GetServerCertificate mit einem AWS SDK oder CLI
- Verwendung GetServiceLastAccessedDetails mit einem AWS SDK oder CLI
- Verwendung GetServiceLastAccessedDetailsWithEntities mit einem AWS SDK oder CLI
- Verwendung GetServiceLinkedRoleDeletionStatus mit einem AWS SDK oder CLI
- Verwendung GetUser mit einem AWS SDK oder CLI
- Verwendung GetUserPolicy mit einem AWS SDK oder CLI
- Verwendung ListAccessKeys mit einem AWS SDK oder CLI
- Verwendung ListAccountAliases mit einem AWS SDK oder CLI
- Verwendung ListAttachedGroupPolicies mit einem AWS SDK oder CLI
- Verwendung ListAttachedRolePolicies mit einem AWS SDK oder CLI
- Verwendung ListAttachedUserPolicies mit einem AWS SDK oder CLI
- Verwendung ListEntitiesForPolicy mit einem AWS SDK oder CLI
- Verwendung ListGroupPolicies mit einem AWS SDK oder CLI
- Verwendung ListGroups mit einem AWS SDK oder CLI

- Verwendung ListGroupsForUser mit einem AWS SDK oder CLI
- Verwendung ListInstanceProfiles mit einem AWS SDK oder CLI
- Verwendung ListInstanceProfilesForRole mit einem AWS SDK oder CLI
- Verwendung ListMfaDevices mit einem AWS SDK oder CLI
- Verwendung ListOpenIdConnectProviders mit einem AWS SDK oder CLI
- Verwendung ListPolicies mit einem AWS SDK oder CLI
- Verwendung ListPolicyVersions mit einem AWS SDK oder CLI
- Verwendung ListRolePolicies mit einem AWS SDK oder CLI
- Verwendung ListRoleTags mit einem AWS SDK oder CLI
- Verwendung ListRoles mit einem AWS SDK oder CLI
- Verwendung ListSAMLProviders mit einem AWS SDK oder CLI
- Verwendung ListServerCertificates mit einem AWS SDK oder CLI
- Verwendung ListSigningCertificates mit einem AWS SDK oder CLI
- Verwendung ListUserPolicies mit einem AWS SDK oder CLI
- Verwendung ListUserTags mit einem AWS SDK oder CLI
- Verwendung ListUsers mit einem AWS SDK oder CLI
- Verwendung ListVirtualMfaDevices mit einem AWS SDK oder CLI
- Verwendung PutGroupPolicy mit einem AWS SDK oder CLI
- Verwendung PutRolePermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung PutRolePolicy mit einem AWS SDK oder CLI
- Verwendung PutUserPermissionsBoundary mit einem AWS SDK oder CLI
- Verwendung PutUserPolicy mit einem AWS SDK oder CLI
- Verwendung RemoveClientIdFromOpenIdConnectProvider mit einem AWS SDK oder CLI
- Verwendung RemoveRoleFromInstanceProfile mit einem AWS SDK oder CLI
- Verwendung RemoveUserFromGroup mit einem AWS SDK oder CLI
- Verwendung ResyncMfaDevice mit einem AWS SDK oder CLI
- Verwendung SetDefaultPolicyVersion mit einem AWS SDK oder CLI
- Verwendung TagRole mit einem AWS SDK oder CLI
- Verwendung TagUser mit einem AWS SDK oder CLI

- Verwendung UntagRole mit einem AWS SDK oder CLI
- Verwendung UntagUser mit einem AWS SDK oder CLI
- Verwendung UpdateAccessKey mit einem AWS SDK oder CLI
- Verwendung UpdateAccountPasswordPolicy mit einem AWS SDK oder CLI
- Verwendung UpdateAssumeRolePolicy mit einem AWS SDK oder CLI
- Verwendung UpdateGroup mit einem AWS SDK oder CLI
- Verwendung UpdateLoginProfile mit einem AWS SDK oder CLI
- Verwendung UpdateOpenIdConnectProviderThumbprint mit einem AWS SDK oder CLI
- Verwendung UpdateRole mit einem AWS SDK oder CLI
- Verwendung UpdateRoleDescription mit einem AWS SDK oder CLI
- Verwendung UpdateSamlProvider mit einem AWS SDK oder CLI
- Verwendung UpdateServerCertificate mit einem AWS SDK oder CLI
- Verwendung UpdateSigningCertificate mit einem AWS SDK oder CLI
- Verwendung UpdateUser mit einem AWS SDK oder CLI
- Verwendung UploadServerCertificate mit einem AWS SDK oder CLI
- Verwendung UploadSigningCertificate mit einem AWS SDK oder CLI

Verwendung **`AddClientIdToOpenIdConnectProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AddClientIdToOpenIdConnectProvider`.

CLI

AWS CLI

So fügen Sie einem Open-ID Connect (OIDC) -Anbieter eine Client-ID (Audience) hinzu

Der folgende `add-client-id-to-open-id-connect-provider` Befehl fügt dem genannten OIDC-Anbieter die Client-ID `my-application-ID` hinzu. `server.example.com`

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
```

```
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den Befehl, um einen OIDC-Anbieter zu erstellen. `create-open-id-connect-provider`

Weitere Informationen finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie in der Befehlsreferenz AddClientIdToOpenIdConnectProvider.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl fügt die Client-ID (oder Audience) **`my-application-ID`** dem vorhandenen OIDC-Anbieter namens hinzu. **`server.example.com`**

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Einzelheiten zur API finden Sie unter AddClientIdToOpenIdConnectProvider AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AddRoleToInstanceProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AddRoleToInstanceProfile`.

CLI

AWS CLI

Um eine Rolle zu einem Instanzprofil hinzuzufügen

Der folgende `add-role-to-instance-profile` Befehl fügt die benannte Rolle `S3Access` dem genannten Instanzprofil hinzu`Webserver`.

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den `create-instance-profile` Befehl, um ein Instanzprofil zu erstellen.

Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AddRoleToInstanceProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Mit diesem Befehl wird die angegebene Rolle einem vorhandenen Instanzprofil mit dem Namen hinzugefügt**`webserver`**. **`S3Access`** Verwenden Sie den **`New-IAMInstanceProfile`** Befehl, um das Instanzprofil zu erstellen. Nachdem Sie das Instanzprofil erstellt und es mithilfe dieses Befehls einer Rolle zugeordnet haben, können Sie es an eine EC2-Instance anhängen. Verwenden Sie dazu das **`New-EC2Instance`** Cmdlet mit dem **`InstanceProfile-Name`** Parameter **`InstanceProfile_Arn`** oder, um die neue Instance zu starten.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Einzelheiten zur API finden Sie unter AddRoleToInstanceProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **AddUserToGroup** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AddUserToGroup`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie AddUserToGroupin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So fügen Sie einen Benutzer einer IAM-Gruppe hinzu

Mit dem folgenden `add-user-to-group`-Befehl wird ein Benutzer mit dem Namen `Bob` zur IAM-Gruppe mit dem Namen `Admins` hinzugefügt.

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Hinzufügen und Entfernen von Benutzern in einer IAM-Benutzergruppe im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie AddUserToGroupin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Mit diesem Befehl wird der angegebene Benutzer **`Bob`** zur Gruppe mit dem Namen hinzugefügt**`Admins`**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Einzelheiten zur API finden Sie unter AddUserToGroup AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AttachGroupPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AttachGroupPolicy`.

CLI

AWS CLI

Um eine verwaltete Richtlinie an eine IAM-Gruppe anzuhängen

Mit dem folgenden `attach-group-policy` Befehl wird die benannte AWS verwaltete Richtlinie `ReadOnlyAccess` an die angegebene IAM-Gruppe angehängt. `Finance`

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen hierzu finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AttachGroupRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die vom Kunden verwaltete Richtlinie mit dem Namen **`TesterPolicy`** der IAM-Gruppe verknüpft. **`Testers`** Die Benutzer in dieser Gruppe sind unmittelbar von den in der Standardversion dieser Richtlinie definierten Berechtigungen betroffen.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Beispiel 2: In diesem Beispiel wird die AWS verwaltete Richtlinie mit dem Namen **`AdministratorAccess`** der IAM-Gruppe angehängt. **`Admins`** Die Benutzer in dieser Gruppe sind unmittelbar von den in der neuesten Version dieser Richtlinie definierten Berechtigungen betroffen.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Einzelheiten zur API finden Sie unter AttachGroupRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AttachRolePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AttachRolePolicy`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Verwalten Sie Rollen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
```

```
        var response = await _IAMService.AttachRolePolicyAsync(new
AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
```

```
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
```

```
    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
<<
                  roleName << std::endl;
    }
```

```
    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So fügen Sie einer IAM-Rolle eine verwaltete Richtlinie an

Mit dem folgenden `attach-role-policy` Befehl wird die benannte AWS verwaltete Richtlinie `ReadOnlyAccess` an die angegebene IAM-Rolle angehängt. `ReadOnlyRole`

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen hierzu finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
```

```
type RoleWrapper struct {
 IamClient *iam.Client
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
```

```
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }
```

```
   public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
       try {
           ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                   .roleName(roleName)
                   .build();

           ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
           List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

           // Ensure that the policy is not attached to this role
           String polArn = "";
           for (AttachedPolicy policy : attachedPolicies) {
               polArn = policy.policyArn();
               if (polArn.compareTo(policyArn) == 0) {
                   System.out.println(roleName + " policy is already attached to
this role.");
                   return;
               }
           }

           AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                   .roleName(roleName)
                   .policyArn(policyArn)
                   .build();

           iam.attachRolePolicy(attachRequest);

           System.out.println("Successfully attached policy " + policyArn +
                   " to role " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       System.out.println("Done");
   }
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Anfügen einer Richtlinie.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
```

```
      }
    });
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }
```

```
        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie im AWS SDK für die Kotlin-API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die AWS verwaltete Richtlinie mit dem Namen **`SecurityAudit`** der IAM-Rolle angehängt. **`CoSecurityAuditors`** Die Benutzer, die diese Rolle übernehmen, sind unmittelbar von den in der neuesten Version dieser Richtlinie definierten Berechtigungen betroffen.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Fügen Sie einer Rolle mithilfe des Boto3-Richtlinienobjekts eine Richtlinie an.

```
def attach_to_role(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
```

```
        raise
```

Fügen Sie einer Rolle mithilfe des Boto3-Rollenobjekts eine Richtlinie an.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS API-Referenz zum SDK for Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispielmodul werden Rollenrichtlinien aufgelistet, erstellt, angehängt und entfernt.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
 #{e.message}")
```

```
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
```

```
    false
  end
end
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie im AWS SDK für die Rust-API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func attachRolePolicy(role: String, policyArn: String) async throws {
    let input = AttachRolePolicyInput(
        policyArn: policyArn,
        roleName: role
    )
    do {
        _ = try await client.attachRolePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Einzelheiten zur API finden Sie unter AttachRoleRichtlinie im AWS SDK für die Swift-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AttachUserPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AttachUserPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern

CLI

AWS CLI

So fügen Sie einem IAM-Benutzer eine verwaltete Richtlinie an

Mit dem folgenden `attach-user-policy` Befehl wird die AWS verwaltete Richtlinie mit `AdministratorAccess` dem Namen des IAM-Benutzers verknüpft. `Alice`

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen hierzu finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AttachUserRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die AWS verwaltete Richtlinie mit dem Namen **`AmazonCognitoPowerUser`** des IAM-Benutzers angehängt. **`Bob`** Der Benutzer ist unmittelbar von den in der neuesten Version dieser Richtlinie definierten Berechtigungen betroffen.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Einzelheiten zur API finden Sie unter AttachUserRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

- Einzelheiten zur API finden Sie unter AttachUserRichtlinie in der AWS API-Referenz zum SDK for Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
  # Attaches a policy to a user
  #
  # @param user_name [String] The name of the user
  # @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_user(user_name, policy_arn)
    @iam_client.attach_user_policy(
      user_name: user_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to user: #{e.message}")
    false
  end
```

- Einzelheiten zur API finden Sie unter AttachUserRichtlinie in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;
```

```
    Ok(())
}
```

- Einzelheiten zur API finden Sie unter AttachUserRichtlinie im AWS SDK für die Rust-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ChangePassword`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ChangePassword`.

CLI

AWS CLI

Um das Passwort für Ihren IAM-Benutzer zu ändern

Um das Passwort für Ihren IAM-Benutzer zu ändern, empfehlen wir, den `--cli-input-json` Parameter zu verwenden, um eine JSON-Datei zu übergeben, die Ihr altes und Ihr neues Passwort enthält. Mit dieser Methode können Sie sichere Passwörter mit nicht alphanumerischen Zeichen verwenden. Es kann schwierig sein, Passwörter mit nicht alphanumerischen Zeichen zu verwenden, wenn Sie sie als Befehlszeilenparameter übergeben. Um den `--cli-input-json` Parameter zu verwenden, verwenden Sie zunächst den `change-password` Befehl mit dem `--generate-cli-skeleton` Parameter, wie im folgenden Beispiel.

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

Mit dem vorherigen Befehl wird eine JSON-Datei mit dem Namen change-password.json erstellt, mit der Sie Ihre alten und neuen Passwörter eingeben können. Die Datei könnte beispielsweise wie folgt aussehen.

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
```

```
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

Verwenden Sie als Nächstes den `change-password` Befehl erneut, um Ihr Passwort zu ändern. Übergeben Sie diesmal den `--cli-input-json` Parameter zur Angabe Ihrer JSON-Datei. Der folgende `change-password` Befehl verwendet den `--cli-input-json` Parameter mit einer JSON-Datei namens change-password.json.

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Dieser Befehl kann nur von IAM-Benutzern aufgerufen werden. Wenn dieser Befehl mit AWS Kontoanmeldeinformationen (Root) aufgerufen wird, gibt der Befehl einen `InvalidUserType` Fehler zurück.

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter So ändert ein AWS IAM-Benutzer sein eigenes Passwort.

- Einzelheiten zur API finden Sie unter ChangePassword AWS CLIBefehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl ändert das Passwort für den Benutzer, der den Befehl ausführt. Dieser Befehl kann nur von IAM-Benutzern aufgerufen werden. Wenn dieser Befehl aufgerufen wird, während Sie mit AWS Kontoanmeldeinformationen (Root) angemeldet sind, gibt der Befehl einen Fehler zurück. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Einzelheiten zur API finden Sie unter ChangePasswordCmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateAccessKey`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateAccessKey`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern
- Verwalten von Zugriffsschlüsseln

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
```

```
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
```

```
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Einzelheiten zur API finden Sie unter CreateAccessKey in AWS CLI Command Reference.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
```

```
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So erstellen Sie einen Zugriffsschlüssel für einen IAM-Benutzer

Mit dem folgenden `create-access-key`-Befehl wird ein Zugriffsschlüssel (Zugriffsschlüssel-ID und geheimer Zugriffsschlüssel) für den IAM-Benutzer mit dem Namen Bob erstellt.

```
aws iam create-access-key \
    --user-name Bob
```

Ausgabe:

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
```

```
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Speichern Sie den geheimen Zugriffsschlüssel an einem sicheren Ort. Bei Verlust kann er nicht wiederhergestellt werden und Sie müssen einen neuen Zugriffsschlüssel erstellen.

Weitere Informationen finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter CreateAccessKey in AWS CLI Command Reference.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
```

```
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }

    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie den Zugriffsschlüssel.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
```

```
    }
}
```

- API-Details finden Sie unter CreateAccessKey in AWS SDK for Kotlin API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein neues Paar aus Zugriffsschlüssel und geheimem Zugriffsschlüssel erstellt und dem Benutzer **`David`** zugewiesen. Stellen Sie sicher, dass Sie die **`SecretAccessKey`** Werte **`AccessKeyId`** und in einer Datei speichern, da Sie die nur zu diesem Zeitpunkt abrufen können. **`SecretAccessKey`** Sie können es später nicht abrufen. Wenn Sie den geheimen Schlüssel verlieren, müssen Sie ein neues Zugriffsschlüsselpaar erstellen.

```
New-IAMAccessKey -UserName David
```

Ausgabe:

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Einzelheiten zur API finden Sie unter Referenz CreateAccesszum Schlüssel im AWS Tools for PowerShell Cmdlet.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Einzelheiten zur API finden Sie unter CreateAccessKey in AWS SDK for Python (Boto3) API-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Dieses Beispielmodul listet Zugriffsschlüssel auf, erstellt, deaktiviert und löscht sie.

```
# Manages access keys for IAM users
class AccessKeyManager
```

```
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
```

```
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter CreateAccessSchlüssel in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Einzelheiten zur API finden Sie unter CreateAccessKey in AWS SDK for Rust API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public func createAccessKey(userName: String) async throws ->
IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie unter CreateAccessKey in AWS SDK for Swift API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateAccountAlias`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateAccountAlias`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So erstellen Sie einen Konto-Alias

Der folgende `create-account-alias` Befehl erstellt den Alias `examplecorp` für Ihr AWS Konto.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Ihre AWS Konto-ID und deren Alias im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie den Konto-Alias.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.

- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- API-Details finden Sie unter CreateAccountAlias im AWS SDK für die Kotlin-API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der Kontoalias für Ihr AWS Konto in geändert**`mycompanyaws`**. Die Adresse der Benutzeranmeldeseite wird in https://mycompanyaws.signin.aws.amazon.com/console geändert. Die ursprüngliche URL, die Ihre Konto-ID-Nummer anstelle des Alias verwendet (https://<accountidnumber>.signin.aws.amazon.com/console), funktioniert weiterhin. Alle zuvor definierten Alias-basierten URLs funktionieren jedoch nicht mehr.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- API-Einzelheiten finden Sie unter CreateAccountAlias in AWS SDK for Python (Boto3) API-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Kontenaliase auflisten, erstellen und löschen.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
```

```
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter CreateAccountAlias in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateGroup`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateGroup`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }
```

- Einzelheiten zur API finden Sie CreateGroupin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So erstellen Sie eine IAM-Gruppe

Mit dem folgenden `create-group`-Befehl wird eine IAM-Gruppe mit dem Namen `Admins` erstellt.

```
aws iam create-group \
    --group-name Admins
```

Ausgabe:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-Benutzergruppen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie CreateGroupin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Einzelheiten zur API finden Sie CreateGroupin der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine neue IAM-Gruppe mit dem Namen **`Developers`** erstellt.

```
New-IAMGroup -GroupName Developers
```

Ausgabe:

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Einzelheiten zur API finden Sie unter CreateGroup AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateInstanceProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateInstanceProfile`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen und Verwalten eines ausfallsicheren Services

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
```

```
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                "\"Version\": \"2012-10-17\"," +
                                "\"Statement\": [{" +
                                     "\"Effect\": \"Allow\"," +
                                     "\"Principal\": {" +
                                     "\"Service\": [" +
                                         "\"ec2.amazonaws.com\"" +
                                     "]" +
                                     "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
```

```
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
```

```
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Einzelheiten zur API finden Sie unter CreateInstanceProfil in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So erstellen Sie ein Instance-Profil

Der folgende `create-instance-profile`-Befehl erstellt ein Instance-Profil mit dem Namen `Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Ausgabe:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Verwenden Sie den `add-role-to-instance-profile`-Befehl, um einem Instance-Profil eine Rolle hinzuzufügen.

Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Gewähren von Berechtigungen für Anwendungen, die in Amazon-EC2-Instances ausgeführt werden im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter CreateInstanceProfil in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
```

```
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
```

- Einzelheiten zur API finden Sie unter CreateInstanceProfil in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein neues IAM-Instanzprofil mit dem Namen **`ProfileForDevEC2Instance`** erstellt. Sie müssen den **`Add-IAMRoleToInstanceProfile`** Befehl separat ausführen, um das Instanzprofil einer vorhandenen IAM-Rolle zuzuordnen, die Berechtigungen für die Instanz bereitstellt. Fügen Sie abschließend das Instance-Profil einer EC2-Instance bei, wenn Sie sie starten. Verwenden Sie dazu das **`New-EC2Instance`** Cmdlet mit dem Parameter oder. **`InstanceProfile_Arn`** **`InstanceProfile_Name`**

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Ausgabe:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Einzelheiten zur API finden Sie unter CreateInstanceProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispiel werden eine Richtlinie, eine Rolle und ein Instance-Profil erstellt und alle miteinander verknüpft.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
```

```
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
```

```
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
```

```
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn
```

- Einzelheiten zur API finden Sie unter CreateInstanceProfil in der AWS API-Referenz zum SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateLoginProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateLoginProfile`.

CLI

AWS CLI

Um ein Passwort für einen IAM-Benutzer zu erstellen

Um ein Passwort für einen IAM-Benutzer zu erstellen, empfehlen wir, den `--cli-input-json` Parameter zu verwenden, um eine JSON-Datei zu übergeben, die das Passwort enthält. Mit dieser Methode können Sie ein sicheres Passwort mit nicht alphanumerischen Zeichen erstellen. Es kann schwierig sein, ein Passwort mit nicht alphanumerischen Zeichen zu erstellen, wenn Sie es als Befehlszeilenparameter übergeben.

Um den `--cli-input-json` Parameter zu verwenden, verwenden Sie zunächst den `create-login-profile` Befehl mit dem `--generate-cli-skeleton` Parameter, wie im folgenden Beispiel.

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

Mit dem vorherigen Befehl wird eine JSON-Datei namens create-login-profile .json erstellt, mit der Sie die Informationen für einen nachfolgenden `create-login-profile` Befehl eingeben können. Beispielsweise:

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

Verwenden Sie als Nächstes den `create-login-profile` Befehl erneut, um ein Passwort für einen IAM-Benutzer zu erstellen. Übergeben Sie diesmal den `--cli-input-json` Parameter zur Angabe Ihrer JSON-Datei. Der folgende `create-login-profile` Befehl

verwendet den `--cli-input-json` Parameter mit einer JSON-Datei namens create-login-profile .json.

```
aws iam create-login-profile \
    --cli-input-json file://create-login-profile.json
```

Ausgabe:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

Wenn das neue Passwort gegen die Kontopasswortrichtlinie verstößt, gibt der Befehl einen `PasswordPolicyViolation` Fehler zurück.

Um das Passwort für einen Benutzer zu ändern, der bereits eines hat, verwenden Sie`update-login-profile`. Verwenden Sie den `update-account-password-policy` Befehl, um eine Passwortrichtlinie für das Konto festzulegen.

Wenn die Kontopasswortrichtlinie dies zulässt, können IAM-Benutzer ihre eigenen Passwörter mithilfe des `change-password` Befehls ändern.

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Passwörter für IAM-Benutzer verwalten.AWS

- Einzelheiten zur API finden Sie unter CreateLoginProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein (temporäres) Passwort für den IAM-Benutzer namens Bob erstellt. Außerdem wird das Kennzeichen gesetzt, dass der Benutzer das Passwort bei der nächsten Anmeldung ändern **Bob** muss.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Ausgabe:

```
CreateDate                    PasswordResetRequired             UserName
----------                    ---------------------             --------
4/14/2015 12:26:30 PM         True                              Bob
```

- Einzelheiten zur API finden Sie unter CreateLoginProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateOpenIdConnectProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateOpenIdConnectProvider`.

CLI

AWS CLI

So erstellen Sie einen OpenID Connect (OIDC) -Anbieter

Um einen OpenID Connect (OIDC) -Anbieter zu erstellen, empfehlen wir, den `--cli-input-json` Parameter zu verwenden, um eine JSON-Datei zu übergeben, die die erforderlichen Parameter enthält. Wenn Sie einen OIDC-Anbieter erstellen, müssen Sie die URL des Anbieters übergeben, und die URL muss mit beginnen. `https://` Es kann schwierig sein, die URL als Befehlszeilenparameter zu übergeben, da der Doppelpunkt (:) und der Schrägstrich (/) in manchen Befehlszeilenumgebungen eine besondere Bedeutung haben. Durch die Verwendung des `--cli-input-json` Parameters wird diese Einschränkung umgangen.

Um den `--cli-input-json` Parameter zu verwenden, verwenden Sie zunächst den `create-open-id-connect-provider` Befehl mit dem `--generate-cli-skeleton` Parameter, wie im folgenden Beispiel.

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

Mit dem vorherigen Befehl wird eine JSON-Datei mit dem Namen create-open-id-connect -provider.json erstellt, mit der Sie die Informationen für einen nachfolgenden Befehl eingeben können. `create-open-id-connect-provider` Beispielsweise:

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

Verwenden Sie als Nächstes den `create-open-id-connect-provider` Befehl erneut, um den OpenID Connect (OIDC) -Anbieter zu erstellen. Übergeben Sie diesmal den `--cli-input-json` Parameter zur Angabe Ihrer JSON-Datei. Der folgende `create-open-id-connect-provider` Befehl verwendet den `--cli-input-json` Parameter mit einer JSON-Datei namens -provider.json. create-open-id-connect

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

Ausgabe:

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

Weitere Informationen zu OIDC-Anbietern finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im IAM-Benutzerhandbuch.AWS

Weitere Informationen zum Abrufen von Fingerabdrücken für einen OIDC-Anbieter finden Sie unter Abrufen des Fingerabdrucks für einen OpenID Connect-Identitätsanbieter im IAM-Benutzerhandbuch.AWS

- Einzelheiten zur API finden Sie unter Anbieter in der Befehlsreferenz. CreateOpen IdConnect AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein IAM-OIDC-Anbieter erstellt, der dem OIDC-kompatiblen Anbieterdienst zugeordnet ist, der sich unter der URL **`https://example.oidcprovider.com`** und der Client-ID befindet. **`my-testapp-1`** Der OIDC-Anbieter stellt den Fingerabdruck bereit. Um den Fingerabdruck zu authentifizieren, folgen Sie den Schritten unter http://docs.aws.amazon.com/IAM/latest/ /identity-providers-oidc-obtain-thumbprint.html. UserGuide

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Ausgabe:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Einzelheiten zur API finden Sie unter AWS Tools for PowerShell Anbieter in der Cmdlet-Referenz. CreateOpen IdConnect

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreatePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreatePolicy`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern
- Verwalten von Richtlinien
- Arbeiten mit der IAM-Policy-Builder-API

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-policy operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"
}
```

- Einzelheiten zur API finden Sie CreatePolicyin der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
```

```
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
                 << "   ]"
                 << "}";

    return stringStream.str();
}
```

- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Beispiel 1: So erstellen Sie eine vom Kunden verwaltete Richtlinie

Mit dem folgenden Befehl wird eine vom Kunden verwaltete Richtlinie mit dem Namen `my-policy` erstellt.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy
```

Bei der Datei `policy` handelt es sich um ein JSON-Dokument im aktuellen `shared`-Ordner, das schreibgeschützten Zugriff auf den Ordner in einem Amazon-S3-Bucket mit dem Namen `my-bucket` gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket/shared/*"
            ]
        }
    ]
}
```

Ausgabe:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
```

```
}
```

Weitere Informationen zur Verwendung von Dateien als Eingabe für Zeichenkettenparameter finden Sie unter Angeben von Parameterwerten für die AWS CLI im AWS CLI-Benutzerhandbuch.

Beispiel 2: So erstellen Sie eine vom Kunden verwaltete Richtlinie mit einer Beschreibung

Mit dem folgenden Befehl wird eine vom Kunden verwaltete Richtlinie mit dem Namen `my-policy` und einer unveränderlichen Beschreibung erstellt:

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for my-bucket"
```

Bei der `policy.json`-Datei handelt es sich um ein JSON-Dokument im aktuellen Ordner, das Zugriff auf alle Put-, List- und Get-Aktionen für einen Amazon-S3-Bucket mit dem Namen `my-bucket` gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Ausgabe:

```
{
```

```
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Weitere Informationen zu identitätsbasierten Richtlinien finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien im AWS -IAM-Benutzerhandbuch.

Beispiel 3: So erstellen Sie eine vom Kunden verwaltete Richtlinie mit Tags

Mit dem folgenden Befehl wird eine vom Kunden verwaltete Richtlinie mit dem Namen `my-policy` mit Tags erstellt. In diesem Beispiel wird das `--tags`-Parameter-Flag mit den folgenden JSON-formatierten Tags verwendet: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternativ kann das `--tags`-Flag mit Tags im Kurzformat `'Key=Department,Value=Accounting Key=Location,Value=Seattle'` verwendet werden.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Bei der `policy.json`-Datei handelt es sich um ein JSON-Dokument im aktuellen Ordner, das Zugriff auf alle Put-, List- und Get-Aktionen für einen Amazon-S3-Bucket mit dem Namen `my-bucket` gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Ausgabe:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {

        ]
    }
}
```

Weitere Informationen zu Tagging-Richtlinien finden Sie unter Tagging von vom Kunden verwalteten Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie CreatePolicyunter AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
```

```
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
```

```
        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
 + result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);

            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie die Richtlinie.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};
```

```
iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"
```

```
    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Einzelheiten zur API finden Sie CreatePolicyin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Action\": \"s3:ListAllMyBuckets\",
                   \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    public function createPolicy(string $policyName, string $policyDocument)
    {
```

```
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Einzelheiten zur API finden Sie CreatePolicyin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird im aktuellen AWS Konto eine neue IAM-Richtlinie mit dem Namen **`MySamplePolicy`** Die Datei **`MySamplePolicy.json`** enthält den Richtlinieninhalt erstellt. Beachten Sie, dass Sie den **`-Raw`** Switch-Parameter verwenden müssen, um die JSON-Richtliniendatei erfolgreich zu verarbeiten.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Einzelheiten zur API finden Sie unter CreatePolicy AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
```

```
        return policy
```

- Einzelheiten zur API finden Sie CreatePolicyin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispielmodul werden Rollenrichtlinien aufgelistet, erstellt, angehängt und entfernt.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
```

```
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter CreatePolicy AWS SDK for RubyAPI-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
```

```
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
    Ok(policy.policy.unwrap())
}
```

- Einzelheiten zur API finden Sie CreatePolicyin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
```

```
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie CreatePolicyin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreatePolicyVersion`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreatePolicyVersion`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Richtlinien

CLI

AWS CLI

So erstellen Sie eine neue Version einer verwalteten Richtlinie

In diesem Beispiel wird eine neue v2-Version der IAM-Richtlinie erstellt, deren ARN `arn:aws:iam::123456789012:policy/MyPolicy` lautet, und sie zur Standardversion gemacht.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Ausgabe:

```
{
```

```
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

Weitere Informationen finden Sie unter Versionsverwaltung von IAM-Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter CreatePolicyVersion in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel erstellt eine neue „v2“ -Version der IAM-Richtlinie, deren ARN lautet, **`arn:aws:iam::123456789012:policy/MyPolicy`** und macht sie zur Standardversion. Die **`NewPolicyVersion.json`** Datei enthält den Inhalt der Richtlinie. Beachten Sie, dass Sie den **`-Raw`** Switch-Parameter verwenden müssen, um die JSON-Richtliniendatei erfolgreich zu verarbeiten.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Ausgabe:

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Einzelheiten zur API finden Sie unter CreatePolicyVersion in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
```

```
        return policy_version
```

- Einzelheiten zur API finden Sie unter CreatePolicyVersion in der AWS API-Referenz zum SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateRole`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Verwalten Sie Rollen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
```

```
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
```

```
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
```

```
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie CreateRolein der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::createIamRole(
```

```
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Beispiel 1: So erstellen Sie eine IAM-Rolle

Mit dem folgenden `create-role`-Befehl wird eine Rolle mit dem Namen `Test-Role` erstellt und ihr eine Vertrauensrichtlinie angefügt.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Ausgabe:

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

Die Vertrauensrichtlinie ist als JSON-Dokument in der Datei Test-Role-Trust-Policy.json definiert. (Der Dateiname und die Erweiterung sind nicht von Bedeutung.) Die Vertrauensrichtlinie muss einen Prinzipal angeben.

Verwenden Sie den `put-role-policy`-Befehl, um die Berechtigungsrichtlinie der Rolle anzufügen.

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

Beispiel 2: So erstellen Sie eine IAM-Rolle mit angegebener maximaler Sitzungsdauer

Mit dem folgenden `create-role`-Befehl wird eine Rolle mit dem Namen `Test-Role` erstellt und eine maximale Sitzungsdauer von 7 200 Sekunden (2 Stunden) festgelegt.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

Ausgabe:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
```

```
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Weitere Informationen finden Sie unter Ändern der maximalen Sitzungsdauer (AWS API) einer Rolle im AWS IAM-Benutzerhandbuch.

Beispiel 3: So erstellen Sie eine IAM-Rolle mit Tags

Mit dem folgenden Befehl wird eine IAM-Rolle `Test-Role` mit Tags erstellt. In diesem Beispiel wird das `--tags`-Parameter-Flag mit den folgenden JSON-formatierten Tags verwendet: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternativ kann das `--tags`-Flag mit Tags im Kurzformat `'Key=Department,Value=Accounting Key=Location,Value=Seattle'` verwendet werden.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Ausgabe:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
```

```
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Weitere Informationen finden Sie unter Taggen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie CreateRolein der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s
```

```
                Where:
                    rolename - The name of the role to create.\s
                    fileLocation - The location of the JSON document that
represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
```

```
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie die -Rolle.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
          },
```

```
          Action: "sts:AssumeRole",
        },
      ],
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
```

```
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine neue Rolle mit dem Namen erstellt **`MyNewRole`** und ihr die in der Datei **`NewRoleTrustPolicy.json`** enthaltene Richtlinie angehängt. Beachten Sie, dass Sie den **`-Raw`** Switch-Parameter verwenden müssen, um die JSON-Richtliniendatei erfolgreich zu verarbeiten. Das in der Ausgabe angezeigte Richtliniendokument ist URL-codiert. In diesem Beispiel wird es mit der **`UrlDecode`** .NET-Methode dekodiert.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Ausgabe:

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                           %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                           %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
```

```
                            %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                            %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Einzelheiten zur API finden Sie unter CreateRole AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- Einzelheiten zur API finden Sie CreateRolein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Creates a role and attaches policies to it.
 #
 # @param role_name [String] The name of the role.
 # @param assume_role_policy_document [Hash] The trust relationship policy
document.
 # @param policy_arns [Array<String>] The ARNs of the policies to attach.
 # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
 def create_role(role_name, assume_role_policy_document, policy_arns)
   response = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: assume_role_policy_document.to_json
   )
   role_arn = response.role.arn

   policy_arns.each do |policy_arn|
     @iam_client.attach_role_policy(
       role_name: role_name,
       policy_arn: policy_arn
     )
   end

   role_arn
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating role: #{e.message}")
   nil
 end
```

- Einzelheiten zur API finden Sie CreateRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Einzelheiten zur API finden Sie CreateRolein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func createRole(name: String, policyDocument: String) async throws ->
String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie CreateRolein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateSAMLProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateSAMLProvider`.

CLI

AWS CLI

So erstellen Sie einen SAML-Anbieter

In diesem Beispiel wird in IAM ein neuer SAML-Anbieter mit dem Namen `MySAMLProvider` erstellt. Es wird durch das SAML-Metadatendokument beschrieben, das sich in der Datei `SAMLMetaData.xml` befindet.

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

Ausgabe:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-SAML-Identitätsanbietern im AWS -IAM-Benutzerhandbuch.

- API-Details finden Sie unter CreateSAMLProvider in der AWS CLI -Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import * as path from "path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Weitere API-Informationen finden Sie unter CreateSAMLProvider in der API-Referenz für AWS SDK for JavaScript .

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine neue SAML-Provider-Entität in IAM erstellt. Sie ist benannt **`MySAMLProvider`** und wird durch das SAML-Metadatendokument in der Datei beschrieben**`SAMLMetaData.xml`**, das separat von der Website des SAML-Dienstanbieters heruntergeladen wurde.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Ausgabe:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- API-Details finden Sie unter CreateSAMLProvider in der Cmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateServiceLinkedRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateServiceLinkedRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
```

```
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So erstellen Sie eine serviceverknüpfte Rolle

Im folgenden `create-service-linked-role` Beispiel wird eine dienstbezogene Rolle für den angegebenen AWS Dienst erstellt und die angegebene Beschreibung angehängt.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Ausgabe:

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
```

```
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter CreateServiceLinkedRole AWS CLIBefehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
```

```
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen einer serviceverknüpften Rolle.

```
import {
```

```
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    } else {
      throw caught;
    }
  }
};
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine servicelinkte Rolle für den Autoscaling-Service erstellt.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Einzelheiten zur API finden Sie unter CreateServiceLinkedRoleCmdlet-Referenz.AWS Tools for PowerShell

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
        raise
```

```
    else:
        return role
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix,)
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
   role_name
 rescue Aws::Errors::ServiceError => e
   @logger.error("Couldn't create service-linked role for #{service_name}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   raise
 end
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public func createServiceLinkedRole(service: String, suffix: String? = nil,
description: String?)
                   async throws -> IAMClientTypes.Role {
       let input = CreateServiceLinkedRoleInput(
           awsServiceName: service,
           customSuffix: suffix,
           description: description
       )
       do {
           let output = try await client.createServiceLinkedRole(input: input)
           guard let role = output.role else {
               throw ServiceHandlerError.noSuchRole
           }
           return role
       } catch {
           throw error
       }
   }
```

- Einzelheiten zur API finden Sie CreateServiceLinkedRolein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateUser`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
```

```
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""
```

```
  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie CreateUserin der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);
```

```
    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Beispiel 1: So erstellen Sie einen IAM-Benutzer

Mit dem folgenden `create-user`-Befehl wird im aktuellen Konto ein IAM-Benutzer mit dem Namen Bob erstellt.

```
aws iam create-user \
    --user-name Bob
```

Ausgabe:

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Einen IAM-Benutzer in Ihrem AWS Konto erstellen.AWS

Beispiel 2: So erstellen Sie einen IAM-Benutzer unter einem angegebenen Pfad

Mit dem folgenden `create-user`-Befehl wird im angegebenen Pfad ein IAM-Benutzer mit dem Namen Bob erstellt.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Ausgabe:

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Weitere Informationen finden Sie unter IAM-Kennungen im AWS -Benutzerhandbuch.

Beispiel 3: So erstellen Sie einen IAM-Benutzer mit Tags

Mit dem folgenden `create-user`-Befehl wird ein IAM-Benutzer mit dem Namen Bob mit Tags erstellt. In diesem Beispiel wird das `--tags`-Parameter-Flag mit den folgenden JSON-formatierten Tags verwendet: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternativ kann das `--tags`-Flag mit Tags im Kurzformat `'Key=Department,Value=Accounting Key=Location,Value=Seattle'` verwendet werden.

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Ausgabe:

```
{
    "User": {
        "Path": "/",
```

```
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Weitere Informationen finden Sie unter Tagging von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

Beispiel 3: So erstellen Sie einen IAM-Benutzer mit einer festgelegten Berechtigungsgrenze

Mit dem folgenden `create-user` Befehl wird ein IAM-Benutzer Bob mit der Berechtigungsgrenze von AmazonS3 erstellt. FullAccess

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Ausgabe:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
```

```
    }
}
```

Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie CreateUserin AWS CLI der Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
```

```
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
        return "";
    }
}
```

- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie den Benutzer.

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Einzelheiten zur API finden Sie CreateUserin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";
```

```
    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
        ]);

        return $result['User'];
    }
```

- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein IAM-Benutzer mit dem Namen **Bob** erstellt. Wenn Bob sich an der AWS Konsole anmelden muss, müssen Sie den Befehl separat ausführen, **New-IAMLoginProfile** um ein Anmeldeprofil mit einem Passwort zu erstellen. Wenn Bob plattformübergreifende CLI-Befehle ausführen AWS PowerShell oder AWS API-Aufrufe tätigen muss, müssen Sie den **New-IAMAccessKey** Befehl separat ausführen, um Zugriffsschlüssel zu erstellen.

```
New-IAMUser -UserName Bob
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Einzelheiten zur API finden Sie unter CreateUser AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- Einzelheiten zur API finden Sie CreateUserin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Creates a user and their login profile
 #
 # @param user_name [String] The name of the user
 # @param initial_password [String] The initial password for the user
 # @return [String, nil] The ID of the user if created, or nil if an error
occurred
 def create_user(user_name, initial_password)
   response = @iam_client.create_user(user_name: user_name)
   @iam_client.wait_until(:user_exists, user_name: user_name)
   @iam_client.create_login_profile(
     user_name: user_name,
     password: initial_password,
     password_reset_required: true
   )
   @logger.info("User '#{user_name}' created successfully.")
   response.user.user_id
 rescue Aws::IAM::Errors::EntityAlreadyExists
   @logger.error("Error creating user '#{user_name}': user already exists.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating user '#{user_name}': #{e.message}")
   nil
 end
```

- Einzelheiten zur API finden Sie CreateUserin der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;

    Ok(response.user.unwrap())
}
```

- Einzelheiten zur API finden Sie CreateUserin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func createUser(name: String) async throws -> String {
        let input = CreateUserInput(
            userName: name
```

```
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie CreateUserin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateVirtualMfaDevice`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateVirtualMfaDevice`.

CLI

AWS CLI

So erstellen Sie ein virtuelles MFA-Gerät

In diesem Beispiel wird ein neues virtuelles MFA-Gerät namens `BobsMFADevice` erstellt. Es erstellt eine Datei, die Bootstrap-Informationen enthält, `QRCode.png` und platziert sie im `C:/` Verzeichnis. Die in diesem Beispiel verwendete Bootstrap-Methode ist. `QRCodePNG`

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

Ausgabe:

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

Weitere Informationen finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie CreateVirtualMfaDevicein der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein neues virtuelles MFA-Gerät erstellt. Die Zeilen 2 und 3 extrahieren den **`Base32StringSeed`** Wert, den das virtuelle MFA-Softwareprogramm benötigt, um ein Konto zu erstellen (als Alternative zum QR-Code). Nachdem Sie das Programm mit dem Wert konfiguriert haben, rufen Sie zwei aufeinanderfolgende Authentifizierungscodes aus dem Programm ab. Verwenden Sie abschließend den letzten Befehl, um das virtuelle MFA-Gerät mit dem IAM-Benutzer zu verknüpfen **`Bob`** und das Konto mit den beiden Authentifizierungscodes zu synchronisieren.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Ausgabe:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Beispiel 2: In diesem Beispiel wird ein neues virtuelles MFA-Gerät erstellt. Die Zeilen 2 und 3 extrahieren den **`QRCodePNG`** Wert und schreiben ihn in eine Datei. Dieses Bild kann vom virtuellen MFA-Softwareprogramm gescannt werden, um ein Konto zu erstellen (als Alternative zur manuellen Eingabe des StringSeed Base32-Werts). Nachdem Sie das Konto in Ihrem

virtuellen MFA-Programm erstellt haben, rufen Sie zwei sequentielle Authentifizierungscodes ab und geben Sie sie in die letzten Befehle ein, um das virtuelle MFA-Gerät mit dem IAM-Benutzer zu verknüpfen **Bob** und das Konto zu synchronisieren.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Ausgabe:

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Einzelheiten zur API finden Sie unter CreateVirtualMfaDeviceCmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeactivateMfaDevice`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeactivateMfaDevice`.

CLI

AWS CLI

Um ein MFA-Gerät zu deaktivieren

Dieser Befehl deaktiviert das virtuelle MFA-Gerät mit dem ARN`arn:aws:iam::210987654321:mfa/BobsMFADevice`, das dem Benutzer zugeordnet ist. Bob

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeactivateMfaDevice in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl deaktiviert das Hardware-MFA-Gerät, das dem Benutzer zugeordnet ist**Bob**, der die Seriennummer besitzt. **123456789012**

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Beispiel 2: Dieser Befehl deaktiviert das virtuelle MFA-Gerät, das dem Benutzer zugeordnet ist**David**, der über den ARN verfügt. **arn:aws:iam::210987654321:mfa/David** Beachten Sie, dass das virtuelle MFA-Gerät nicht aus dem Konto gelöscht wird. Das virtuelle Gerät ist immer noch vorhanden und erscheint in der Ausgabe des **Get-IAMVirtualMFADevice** Befehls. Bevor Sie ein neues virtuelles MFA-Gerät für denselben Benutzer erstellen können, müssen Sie das alte mit dem **Remove-IAMVirtualMFADevice** Befehl löschen.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Einzelheiten zur API finden Sie unter DeactivateMfaDevice in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **DeleteAccessKey** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteAccessKey`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern
- Verwalten von Zugriffsschlüsseln

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
```

```
  echo "  -k access_key   The access key to delete."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:k:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    k) access_key="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Einzelheiten zur API finden Sie unter DeleteAccessKey in AWS CLI Command Reference.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
```

```
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So löschen Sie einen Zugriffsschlüssel für einen IAM-Benutzer

Mit dem folgenden `delete-access-key`-Befehl wird der angegebene Zugriffsschlüssel (Zugriffsschlüssel-ID und geheimer Zugriffsschlüssel) für den IAM-Benutzer mit dem Namen Bob gelöscht.

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den `list-access-keys`-Befehl, um die für einen IAM-Benutzer definierten Zugriffsschlüssel aufzulisten.

Weitere Informationen finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteAccessKey in AWS CLI Command Reference.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie den Zugriffsschlüssel.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
```

```
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
```

```
}
```

- API-Details finden Sie unter DeleteAccessKey in AWS SDK for Kotlin API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das AWS Zugriffsschlüsselpaar mit der Schlüssel-ID des **AKIAIOSFODNN7EXAMPLE** angegebenen Benutzers gelöscht. **Bob**

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Einzelheiten zur API finden Sie unter DeleteAccessKey in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
```

```
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Einzelheiten zur API finden Sie unter DeleteAccessKey in AWS SDK for Python (Boto3) API-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Dieses Beispielmodul listet Zugriffsschlüssel auf, erstellt, deaktiviert und löscht sie.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
```

```
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
```

```
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter DeleteAccessSchlüssel in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
```

```
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
        }
    }
    Ok(())
}
```

- Einzelheiten zur API finden Sie unter DeleteAccessKey in AWS SDK for Rust API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }
```

```
        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie unter DeleteAccessKey in AWS SDK for Swift API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteAccountAlias`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteAccountAlias`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie unter DeleteAccountAlias in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So löschen Sie einen Konto-Alias

Mit dem folgenden `delete-account-alias`-Befehl wird der Alias `mycompany` für das aktuelle Konto entfernt.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Ihre AWS Konto-ID und deren Alias im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteAccountAlias in der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Einzelheiten zur API finden Sie unter DeleteAccountAlias in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie den Konto-Alias.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DeleteAccountAlias in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
```

```
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DeleteAccountAlias in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- API-Details finden Sie unter DeleteAccountAlias im AWS SDK für die Kotlin-API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der Account-Alias aus Ihrem entfernt AWS-Konto. Die Benutzeranmeldeseite mit dem Alias unter https://mycompanyaws.signin.aws.amazon.com/console funktioniert nicht mehr. Sie müssen stattdessen die ursprüngliche URL mit Ihrer

AWS-Konto ID-Nummer unter https://.signin.aws.amazon.com/console verwenden. <accountidnumber>

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Einzelheiten zur API finden Sie unter Alias in der Cmdlet-Referenz. DeleteAccount AWS Tools for PowerShell

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- API-Einzelheiten finden Sie unter DeleteAccountAlias in AWS SDK for Python (Boto3) API-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Kontenaliase auflisten, erstellen und löschen.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter DeleteAccountAlias in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteAccountPasswordPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteAccountPasswordPolicy`.

CLI

AWS CLI

Um die Passwortrichtlinie für das aktuelle Konto zu löschen

Mit dem folgenden `delete-account-password-policy` Befehl wird die Kennwortrichtlinie für das aktuelle Konto entfernt.

```
aws iam delete-account-password-policy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Festlegen einer Kontopasswortrichtlinie für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteAccountPasswordPolicyin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Kennwortrichtlinie für gelöscht AWS-Konto und alle Werte auf ihre ursprünglichen Standardwerte zurückgesetzt. Wenn derzeit keine Kennwortrichtlinie existiert, wird die folgende Fehlermeldung angezeigt: Die Kontorichtlinie mit dem Namen PasswordPolicy kann nicht gefunden werden.

```
Remove-IAMAccountPasswordPolicy
```

- Einzelheiten zur API finden Sie unter DeleteAccountPasswordPolicy AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteGroup`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteGroup`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteGroupin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So löschen Sie eine IAM-Gruppe

Mit dem folgenden `delete-group`-Befehl wird eine IAM-Gruppe mit dem Namen `MyTestGroup` gelöscht.

```
aws iam delete-group \
    --group-name MyTestGroup
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Löschen einer IAM-Benutzergruppe im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteGroupin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Einzelheiten zur API finden Sie DeleteGroupin der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die IAM-Gruppe mit dem Namen gelöscht. **`MyTestGroup`** Mit dem ersten Befehl werden alle IAM-Benutzer entfernt, die Mitglieder der Gruppe sind, und mit dem zweiten Befehl wird die IAM-Gruppe gelöscht. Beide Befehle funktionieren ohne Aufforderung zur Bestätigung.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Einzelheiten zur API finden Sie unter DeleteGroup AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteGroupPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteGroupPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
 policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie unter DeleteGroupRichtlinie in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So löschen Sie eine Richtlinie aus einer IAM-Gruppe

Mit dem folgenden `delete-group-policy`-Befehl wird die Richtlinie mit dem Namen `ExamplePolicy` aus der Gruppe mit dem Namen `Admins` gelöscht.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den `list-group-policies`-Befehl, um die einer Gruppe zugeordneten Richtlinien anzuzeigen.

Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteGroupRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Inline-Richtlinie mit dem Namen **`TesterPolicy`** aus der IAM-Gruppe **`Testers`** entfernt. Die Benutzer in dieser Gruppe verlieren sofort die in dieser Richtlinie definierten Berechtigungen.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Einzelheiten zur API finden Sie unter DeleteGroupRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteInstanceProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteInstanceProfile`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen und Verwalten eines ausfallsicheren Services

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }
```

```
            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- Einzelheiten zur API finden Sie unter DeleteInstanceProfil in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So löschen Sie ein Instance-Profil

Mit dem folgenden `delete-instance-profile`-Befehl wird das Instance-Profil mit dem Namen `ExampleInstanceProfile` gelöscht.

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwenden von Instance-Profilen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteInstanceProfil in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    const client = new IAMClient({});
    await client.send(
      new DeleteInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
```

- Einzelheiten zur API finden Sie unter DeleteInstanceProfil in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das angegebene EC2-Instanzprofil gelöscht. **`MyAppInstanceProfile`** Mit dem ersten Befehl werden alle Rollen vom Instanzprofil getrennt, und mit dem zweiten Befehl wird das Instanzprofil gelöscht.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Einzelheiten zur API finden Sie unter DeleteInstanceProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispiel wird die Rolle aus dem Instance-Profil entfernt, alle mit der Rolle verknüpften Richtlinien getrennt und alle Ressourcen gelöscht.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"
```

```
   def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )
```

- Einzelheiten zur API finden Sie unter DeleteInstanceProfil in der AWS API-Referenz zum SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteLoginProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteLoginProfile`.

CLI

AWS CLI

Um ein Passwort für einen IAM-Benutzer zu löschen

Der folgende `delete-login-profile` Befehl löscht das Passwort für den IAM-Benutzer mit dem Namen. `Bob`

```
aws iam delete-login-profile \
    --user-name Bob
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Passwörter für IAM-Benutzer verwalten.AWS

- Einzelheiten zur API finden Sie unter DeleteLoginProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Anmeldeprofil des IAM-Benutzers mit dem Namen gelöscht. **`Bob`** Dadurch wird verhindert, dass sich der Benutzer an der Konsole anmeldet. AWS Es verhindert nicht, dass der Benutzer AWS CLI- PowerShell oder API-Aufrufe mit AWS Zugriffsschlüsseln ausführt, die möglicherweise noch mit dem Benutzerkonto verknüpft sind.

```
Remove-IAMLoginProfile -UserName Bob
```

- Einzelheiten zur API finden Sie unter DeleteLoginProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteOpenIdConnectProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteOpenIdConnectProvider`.

CLI

AWS CLI

So löschen Sie einen IAM OpenID Connect-Identitätsanbieter

In diesem Beispiel wird der IAM-OIDC-Anbieter gelöscht, der eine Verbindung zum Anbieter herstellt. `example.oidcprovider.com`

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteOpenIdConnectAnbieter in der Befehlsreferenz.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der IAM-OIDC-Anbieter gelöscht, der eine Verbindung zum Anbieter herstellt. **`example.oidcprovider.com`** Stellen Sie sicher, dass Sie alle Rollen aktualisieren oder löschen, die im **`Principal`** Element der Vertrauensrichtlinie der Rolle auf diesen Anbieter verweisen.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Einzelheiten zur API finden Sie unter DeleteOpenIdConnectAnbieter in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeletePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeletePolicy`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern
- Verwalten von Richtlinien

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
```

```
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeletePolicyin der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_policy
```

```
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);
```

```
    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So löschen Sie eine IAM-Richtlinie

In diesem Beispiel wird die Richtlinie, deren ARN `arn:aws:iam::123456789012:policy/MySamplePolicy` lautet, gelöscht.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeletePolicyin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie die Richtlinie.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Einzelheiten zur API finden Sie DeletePolicyin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der API-Referenz zum AWS SDK für Kotlin.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Richtlinie gelöscht, deren ARN lautet**`arn:aws:iam::123456789012:policy/MySamplePolicy`**. Bevor Sie die Richtlinie löschen können, müssen Sie zuerst alle Versionen mit Ausnahme der Standardversion löschen, indem Sie Folgendes ausführen**`Remove-IAMPolicyVersion`**. Sie müssen die Richtlinie auch von allen IAM-Benutzern, -Gruppen oder -Rollen trennen.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Beispiel 2: In diesem Beispiel wird eine Richtlinie gelöscht, indem zuerst alle nicht standardmäßigen Richtlinienversionen gelöscht werden, sie von allen angehängten IAM-Entitäten getrennt und schließlich die Richtlinie selbst gelöscht wird. In der ersten Zeile wird das Richtlinienobjekt abgerufen. In der zweiten Zeile werden alle Richtlinienversionen, die nicht als Standardversion gekennzeichnet sind, in einer Sammlung abgerufen und anschließend alle Richtlinien in der Sammlung gelöscht. In der dritten Zeile werden alle IAM-Benutzer, -Gruppen und -Rollen abgerufen, denen die Richtlinie zugeordnet ist. In den Zeilen vier bis sechs wird die Richtlinie von jeder angehängten Entität getrennt. In der letzten Zeile wird dieser Befehl verwendet, um die verwaltete Richtlinie sowie die verbleibende Standardversion zu entfernen. Das Beispiel enthält den **`-Force`** Switch-Parameter in jeder Zeile, die ihn benötigt, um Bestätigungsaufforderungen zu unterdrücken.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Einzelheiten zur API finden Sie unter DeletePolicy AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Einzelheiten zur API finden Sie DeletePolicyin AWS SDK for Python (Boto3) API Reference.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
```

```
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
    let input = DeletePolicyInput(
        policyArn: policy.arn
    )
    do {
        _ = try await iamClient.deletePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Einzelheiten zur API finden Sie DeletePolicyin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeletePolicyVersion`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeletePolicyVersion`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Verwalten von Richtlinien
- Zurücksetzen einer Richtlinienversion

CLI

AWS CLI

Um eine Version einer verwalteten Richtlinie zu löschen

In diesem Beispiel wird die als identifizierte Version `v2` aus der Richtlinie gelöscht, deren ARN lautet`arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeletePolicyVersion in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die als identifizierte Version **`v2`** aus der Richtlinie gelöscht, deren ARN lautet**`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Beispiel 2: In diesem Beispiel wird eine Richtlinie gelöscht, indem zuerst alle nicht standardmäßigen Richtlinienversionen und dann die Richtlinie selbst gelöscht werden. In der ersten Zeile wird das Richtlinienobjekt abgerufen. In der zweiten Zeile werden alle Richtlinienversionen, die nicht als Standard gekennzeichnet sind, in einer Sammlung abgerufen und anschließend mit diesem Befehl alle Richtlinien in der Sammlung gelöscht. In der letzten Zeile werden die Richtlinie selbst sowie die verbleibende Standardversion entfernt. Beachten Sie, dass Sie zum erfolgreichen Löschen einer verwalteten Richtlinie auch die Richtlinie mithilfe der **`Unregister-IAMRolePolicy`** Befehle, und von allen Benutzern, Gruppen oder Rollen trennen müssen. **`Unregister-IAMUserPolicy`** **`Unregister-IAMGroupPolicy`** Sehen Sie sich das Beispiel für das **`Remove-IAMPolicy`** Cmdlet an.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Einzelheiten zur API finden Sie unter DeletePolicyVersion in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteRole`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Verwalten Sie Rollen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteRolein der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
```

```
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
```

```
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- Einzelheiten zur API finden Sie DeleteRolein der AWS CLI Befehlsreferenz.

CLI

AWS CLI

So löschen Sie eine IAM-Rolle

Mit dem folgenden `delete-role`-Befehl wird die Rolle mit dem Namen `Test-Role` entfernt.

```
aws iam delete-role \
    --role-name Test-Role
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Bevor Sie eine Rolle löschen können, müssen Sie die Rolle aus allen Instance-Profilen entfernen (`remove-role-from-instance-profile`), alle verwalteten Richtlinien entfernen (`detach-role-policy`) und alle eingebundenen Richtlinien, die der Rolle angefügt sind (`delete-role-policy`), löschen.

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen und Verwenden von Instance-Profilen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteRolein der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie DeleteRolein der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie die Rolle.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
```

```
};
```

- Einzelheiten zur API finden Sie DeleteRolein der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die angegebene Rolle **`MyNewRole`** aus dem aktuellen IAM-Konto gelöscht. Bevor Sie die Rolle löschen können, müssen Sie zunächst den **`Unregister-IAMRolePolicy`** Befehl verwenden, um alle verwalteten Richtlinien zu trennen. Inline-Richtlinien werden zusammen mit der Rolle gelöscht.

```
Remove-IAMRole -RoleName MyNewRole
```

Beispiel 2: In diesem Beispiel werden alle verwalteten Richtlinien von der genannten Rolle getrennt **`MyNewRole`** und anschließend die Rolle gelöscht. In der ersten Zeile werden alle verwalteten Richtlinien, die der Rolle zugeordnet sind, als Sammlung abgerufen und anschließend jede Richtlinie in der Sammlung von der Rolle getrennt. In der zweiten Zeile wird die Rolle selbst gelöscht. Inline-Richtlinien werden zusammen mit der Rolle gelöscht.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Einzelheiten zur API finden Sie unter DeleteRole AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Einzelheiten zur API finden Sie DeleteRolein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    begin
      # Detach and delete attached policies
      @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
        response.attached_policies.each do |policy|
          @iam_client.detach_role_policy({
                                           role_name: role_name,
                                           policy_arn: policy.policy_arn
                                         })
```

```
          # Check if the policy is a customer managed policy (not AWS managed)
          unless policy.policy_arn.include?("aws:policy/")
            @iam_client.delete_policy({ policy_arn: policy.policy_arn })
            @logger.info("Deleted customer managed policy
 #{policy.policy_name}.")
          end
        end
      end

      # Delete the role
      @iam_client.delete_role({ role_name: role_name })
      @logger.info("Deleted role #{role_name}.")
    rescue Aws::IAM::Errors::ServiceError => e
      @logger.error("Couldn't detach policies and delete role #{role_name}.
 Here's why:")
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Einzelheiten zur API finden Sie DeleteRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
```

```
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Einzelheiten zur API finden Sie DeleteRolein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie DeleteRolein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteRolePermissionsBoundary`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteRolePermissionsBoundary`.

CLI

AWS CLI

Um eine Berechtigungsgrenze aus einer IAM-Rolle zu löschen

Im folgenden `delete-role-permissions-boundary` Beispiel wird die Berechtigungsgrenze für die angegebene IAM-Rolle gelöscht. Verwenden Sie den Befehl, um einer Rolle eine Berechtigungsgrenze zuzuweisen. `put-role-permissions-boundary`

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteRolePermissionsBoundaryin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel zeigt, wie die mit einer IAM-Rolle verbundene Berechtigungsgrenze entfernt wird.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Einzelheiten zur API finden Sie unter DeleteRolePermissionsBoundary AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteRolePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteRolePolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie unter DeleteRoleRichtlinie in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So entfernen Sie eine Richtlinie aus einer IAM-Rolle

Mit dem folgenden `delete-role-policy`-Befehl wird die Richtlinie mit dem Namen `ExamplePolicy` aus der Rolle mit dem Namen `Test-Role` entfernt.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Ändern einer Rolle im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteRoleRichtlinie in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
```

```
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- Einzelheiten zur API finden Sie unter DeleteRoleRichtlinie in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Inline-Richtlinie gelöscht**S3AccessPolicy**, die in die IAM-Rolle eingebettet ist. **S3BackupRole**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Einzelheiten zur API finden Sie unter DeleteRoleRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **DeleteSAMLProvider** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteSAMLProvider`.

CLI

AWS CLI

So löschen Sie einen SAML-Anbieter

In diesem Beispiel wird der IAM SAML 2.0-Anbieter gelöscht, dessen ARN `arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider` lautet.

```
aws iam delete-saml-provider \
```

```
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Erstellen von IAM-SAML-Identitätsanbietern im AWS -IAM-Benutzerhandbuch.

- API-Details finden Sie unter DeleteSAMLProvider in der AWS CLI -Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Weitere API-Informationen finden Sie unter DeleteSAMLProvider in der API-Referenz für AWS SDK for JavaScript .

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der IAM SAML 2.0-Anbieter gelöscht, dessen ARN lautet. **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- API-Einzelheiten finden Sie unter DeleteSAMLProvider in der Cmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteServerCertificate`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteServerCertificate`.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeleteServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    const auto outcome = iam.DeleteServerCertificate(request);
    bool result = true;
```

```
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error deleting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- Einzelheiten zur API finden Sie unter DeleteServerZertifikat in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um ein Serverzertifikat aus Ihrem AWS Konto zu löschen

Mit dem folgenden `delete-server-certificate` Befehl wird das angegebene Serverzertifikat aus Ihrem AWS Konto entfernt.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den `list-server-certificates` Befehl, um die in Ihrem AWS Konto verfügbaren Serverzertifikate aufzulisten.

Weitere Informationen finden Sie unter Verwaltung von Serverzertifikaten in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteServerZertifikat in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie ein Serverzertifikat.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DeleteServerZertifikat in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DeleteServerZertifikat in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Serverzertifikat mit dem Namen **`MyServerCert`** gelöscht.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Einzelheiten zur API finden Sie unter DeleteServerCertificate in AWS Tools for PowerShell Cmdlet Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Serverzertifikate auflisten, aktualisieren und löschen.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates
```

```
    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter DeleteServerZertifikat in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteServiceLinkedRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteServiceLinkedRole`.

CLI

AWS CLI

So löschen Sie eine serviceverknüpfte Rolle

Im folgenden `delete-service-linked-role`-Beispiel wird die angegebene serviceverknüpfte Rolle, die Sie nicht mehr benötigen, gelöscht. Der Löschvorgang erfolgt asynchron. Sie können den Status des Löschvorgangs mithilfe des `get-service-linked-role-deletion-status`-Befehls überprüfen und bestätigen, wann der Vorgang abgeschlossen ist.

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Ausgabe:

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteServiceLinkedRolein der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie DeleteServiceLinkedRolein der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Einzelheiten zur API finden Sie DeleteServiceLinkedRolein der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wurde die mit dem Dienst verknüpfte Rolle gelöscht. Bitte beachten Sie, dass dieser Befehl zu einem Fehler führt, wenn der Dienst diese Rolle immer noch verwendet.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Einzelheiten zur API finden Sie unter DeleteServiceLinkedRole AWS Tools for PowerShellCmdlet-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Deletes a service-linked role.
#
# @param role_name [String] The name of the role to delete.
def delete_service_linked_role(role_name)
  response = @iam_client.delete_service_linked_role(role_name: role_name)
  task_id = response.deletion_task_id
  check_deletion_status(role_name, task_id)
rescue Aws::Errors::ServiceError => e
  handle_deletion_error(e, role_name)
end

private

# Checks the deletion status of a service-linked role
#
# @param role_name [String] The name of the role being deleted
# @param task_id [String] The task ID for the deletion process
def check_deletion_status(role_name, task_id)
  loop do
    response = @iam_client.get_service_linked_role_deletion_status(
      deletion_task_id: task_id)
    status = response.status
    @logger.info("Deletion of #{role_name} #{status}.")
    break if %w[SUCCEEDED FAILED].include?(status)
    sleep(3)
  end
end

# Handles deletion error
#
# @param e [Aws::Errors::ServiceError] The error encountered during deletion
# @param role_name [String] The name of the role attempted to delete
```

```
  def handle_deletion_error(e, role_name)
    unless e.code == "NoSuchEntity"
      @logger.error("Couldn't delete #{role_name}. Here's why:")
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Einzelheiten zur API finden Sie DeleteServiceLinkedRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Einzelheiten zur API finden Sie DeleteServiceLinkedRolein der API-Referenz zum AWS SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteSigningCertificate`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteSigningCertificate`.

CLI

AWS CLI

Um ein Signaturzertifikat für einen IAM-Benutzer zu löschen

Der folgende `delete-signing-certificate` Befehl löscht das angegebene Signaturzertifikat für den genannten IAM-Benutzer. `Bob`

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den Befehl, um die ID für ein Signaturzertifikat abzurufen. `list-signing-certificates`

Weitere Informationen finden Sie unter Signaturzertifikate verwalten im Amazon EC2 EC2-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DeleteSigningCertificate in AWS CLI Command Reference.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Signaturzertifikat mit der ID des IAM-Benutzers mit **`Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU`** dem Namen gelöscht. **`Bob`**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Einzelheiten zur API finden Sie unter DeleteSigningCertificate in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteUser`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers
- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
```

```
    }
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
```

```
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
```

```
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So löschen Sie einen IAM-Benutzer

Mit dem folgenden `delete-user`-Befehl wird der IAM-Benutzer mit dem Namen Bob aus dem aktuellen Konto entfernt.

```
aws iam delete-user \
    --user-name Bob
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Löschen eines IAM-Benutzers im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteUserin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie den Benutzer.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
```

```
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteIAMUser(userNameVal: String) {
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Einzelheiten zur API finden Sie DeleteUserin der API-Referenz zum AWS SDK für Kotlin.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der IAM-Benutzer mit dem Namen gelöscht. **Bob**

```
Remove-IAMUser -UserName Bob
```

Beispiel 2: In diesem Beispiel wird der angegebene IAM-Benutzer **Theresa** zusammen mit allen Elementen gelöscht, die zuerst gelöscht werden müssen.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }
```

```
# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Einzelheiten zur API finden Sie unter DeleteUser AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise
```

- Einzelheiten zur API finden Sie DeleteUserin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- Einzelheiten zur API finden Sie DeleteUserin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func deleteUser(user: IAMClientTypes.User) async throws {
    let input = DeleteUserInput(
        userName: user.userName
    )
    do {
        _ = try await iamClient.deleteUser(input: input)
    } catch {
        throw error
    }
}
```

- Einzelheiten zur API finden Sie DeleteUserin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteUserPermissionsBoundary`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteUserPermissionsBoundary`.

CLI

AWS CLI

Um eine Berechtigungsgrenze für einen IAM-Benutzer zu löschen

Im folgenden `delete-user-permissions-boundary` Beispiel wird die Berechtigungsgrenze gelöscht, die dem IAM-Benutzer mit dem Namen zugewiesen ist. `intern` Verwenden Sie den Befehl, um einem Benutzer eine Berechtigungsgrenze zuzuweisen. `put-user-permissions-boundary`

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie DeleteUserPermissionsBoundaryin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel zeigt, wie die einem IAM-Benutzer zugeordnete Berechtigungsgrenze entfernt wird.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Einzelheiten zur API finden Sie unter DeleteUserPermissionsBoundary AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteUserPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteUserPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So entfernen Sie eine Richtlinie von einem IAM-Benutzer

Mit dem folgenden `delete-user-policy`-Befehl wird die angegebene Richtlinie vom IAM-Benutzer mit dem Namen Bob entfernt.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den `list-user-policies`-Befehl, um eine Liste der Richtlinien für einen IAM-Benutzer abzurufen.

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Einen IAM-Benutzer in Ihrem AWS Konto erstellen.AWS

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
```

```
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie in der AWS SDK for Go API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die angegebene Inline-Richtlinie gelöscht**AccessToEC2Policy**, die in den IAM-Benutzer mit dem Namen eingebettet ist. **Bob**

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Beispiel 2: In diesem Beispiel werden alle Inline-Richtlinien gefunden, die in den IAM-Benutzernamen eingebettet sind, **Theresa** und sie werden dann gelöscht.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie in AWS Tools for PowerShell der Cmdlet-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie im AWS SDK für die Rust-API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
```

```
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie unter DeleteUserRichtlinie im AWS SDK für die Swift-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteVirtualMfaDevice`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteVirtualMfaDevice`.

CLI

AWS CLI

So entfernen Sie ein virtuelles MFA-Gerät

Mit dem folgenden `delete-virtual-mfa-device` Befehl wird das angegebene MFA-Gerät aus dem aktuellen Konto entfernt.

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Deaktivierung von MFA-Geräten im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter Befehlsreferenz DeleteVirtualMfaDevice.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das virtuelle IAM-MFA-Gerät gelöscht, dessen ARN lautet. **`arn:aws:iam::123456789012:mfa/bob`**

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Beispiel 2: In diesem Beispiel wird geprüft, ob der IAM-Benutzerin Theresa ein MFA-Gerät zugewiesen wurde. Wenn eines gefunden wird, ist das Gerät für den IAM-Benutzer deaktiviert. Wenn das Gerät virtuell ist, wird es ebenfalls gelöscht.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Einzelheiten zur API finden Sie unter DeleteVirtualMfaDevice AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DetachGroupPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DetachGroupPolicy`.

CLI

AWS CLI

Um eine Richtlinie von einer Gruppe zu trennen

In diesem Beispiel wird die verwaltete Richtlinie mit dem ARN `arn:aws:iam::123456789012:policy/TesterAccessPolicy` aus der aufgerufenen Gruppe entferntTesters.

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwaltung von IAM-Benutzergruppen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DetachGroupRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die verwaltete Gruppenrichtlinie, deren ARN stammt, **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** von der genannten **`Testers`** Gruppe getrennt.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Beispiel 2: In diesem Beispiel werden alle verwalteten Richtlinien gefunden, die der genannten Gruppe zugeordnet sind, **`Testers`** und sie werden von der Gruppe getrennt.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Einzelheiten zur API finden Sie unter DetachGroupRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DetachRolePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DetachRolePolicy`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an
- Verwalten Sie Rollen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
 "
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So trennen Sie eine Richtlinie von einer Rolle

In diesem Beispiel wird die verwaltete Richtlinie mit dem ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` aus der Rolle mit dem Namen `FedTesterRole` entfernt.

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Ändern einer Rolle im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }

    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
```

```
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Trennen Sie die Richtlinie.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });
```

```
  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
```

```
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

```
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie im AWS SDK für die Kotlin-API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die verwaltete Gruppenrichtlinie, deren ARN stammt, **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** von der genannten **`FedTesterRole`** Rolle getrennt.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Beispiel 2: In diesem Beispiel werden alle verwalteten Richtlinien gefunden, die der genannten Rolle zugeordnet sind, **`FedTesterRole`** und sie werden von der Rolle getrennt.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Trennen Sie eine Richtlinie von einer Rolle mithilfe des Boto3-Richtlinienobjekts.

```
def detach_from_role(role_name, policy_arn):
```

```
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Trennen Sie eine Richtlinie von einer Rolle mithilfe des Boto3-Rollenobjekts.

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS API-Referenz zum SDK for Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispielmodul werden Rollenrichtlinien aufgelistet, erstellt, angehängt und entfernt.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;
```

```
    Ok(())
}
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie im AWS SDK für die Rust-API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
IAMClientTypes.Role) async throws {
        let input = DetachRolePolicyInput(
            policyArn: policy.arn,
            roleName: role.roleName
        )

        do {
            _ = try await iamClient.detachRolePolicy(input: input)
        } catch {
            throw error
        }
   }
```

- Einzelheiten zur API finden Sie unter DetachRoleRichtlinie im AWS SDK für die Swift-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DetachUserPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DetachUserPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern

CLI

AWS CLI

So trennen Sie eine Richtlinie von einem Benutzer

In diesem Beispiel wird die verwaltete Richtlinie mit dem ARN `arn:aws:iam::123456789012:policy/TesterPolicy` vom Benutzer `Bob` entfernt.

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Änderung der Berechtigungen für einen IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DetachUserRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die verwaltete Richtlinie, deren ARN stammt, **`arn:aws:iam::123456789012:policy/TesterPolicy`** von dem IAM-Benutzer mit dem Namen getrennt. **`Bob`**

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Beispiel 2: In diesem Beispiel werden alle verwalteten Richtlinien gefunden, die dem IAM-Benutzer mit dem Namen zugeordnet sind, **`Theresa`** und diese Richtlinien werden vom Benutzer getrennt.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Einzelheiten zur API finden Sie unter DetachUserRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
```

```
            logger.info("Detached policy %s from user %s.", policy_arn, user_name)
        except ClientError:
            logger.exception(
                "Couldn't detach policy %s from user %s.", policy_arn, user_name
            )
            raise
```

- Einzelheiten zur API finden Sie unter DetachUserRichtlinie in der AWS API-Referenz zum SDK for Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
   )
   @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
   true
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Error detaching policy: Policy or user does not exist.")
   false
 rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error detaching policy from user '#{user_name}':
 #{e.message}")
    false
  end
```

- Einzelheiten zur API finden Sie unter DetachUserRichtlinie in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Einzelheiten zur API finden Sie unter DetachUserRichtlinie im AWS SDK für die Rust-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`EnableMfaDevice`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`EnableMfaDevice`.

CLI

AWS CLI

So aktivieren Sie ein MFA-Gerät

Nachdem Sie den `create-virtual-mfa-device` Befehl verwendet haben, um ein neues virtuelles MFA-Gerät zu erstellen, können Sie das MFA-Gerät einem Benutzer zuweisen. Im folgenden `enable-mfa-device` Beispiel wird dem Benutzer das MFA-Gerät mit der Seriennummer `arn:aws:iam::210987654321:mfa/BobsMFADevice` zugewiesen. Bob Der Befehl synchronisiert das Gerät auch mit, AWS indem er die ersten beiden Codes nacheinander vom virtuellen MFA-Gerät einfügt.

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Aktivieren eines Geräts mit virtueller Multi-Faktor-Authentifizierung (MFA) im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter EnableMfaDevice in AWS CLI Command Reference.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl aktiviert das Hardware-MFA-Gerät mit der Seriennummer **`987654321098`** und ordnet das Gerät dem Benutzer **`Bob`** zu. Er enthält nacheinander die ersten beiden Codes des Geräts.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Beispiel 2: In diesem Beispiel wird ein virtuelles MFA-Gerät erstellt und aktiviert. Der erste Befehl erstellt das virtuelle Gerät und gibt die Objektdarstellung des Geräts in der Variablen **`$MFADevice`** zurück. Sie können die **`QRCodePng`** Eigenschaften **`.Base32StringSeed`** oder verwenden, um die Softwareanwendung des Benutzers zu konfigurieren. Mit dem letzten Befehl wird das Gerät dem Benutzer zugewiesen **`David`** und das Gerät anhand seiner Seriennummer identifiziert. Der Befehl synchronisiert das Gerät auch mit, AWS indem er die ersten beiden Codes nacheinander vom virtuellen MFA-Gerät einfügt.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Einzelheiten zur API finden Sie unter EnableMfaDevice in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GenerateCredentialReport`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GenerateCredentialReport`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

CLI

AWS CLI

So erstellen Sie einen Bericht zu Anmeldeinformationen

Im folgenden Beispiel wird versucht, einen Anmeldeinformationsbericht für das AWS Konto zu generieren.

```
aws iam generate-credential-report
```

Ausgabe:

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Weitere Informationen finden Sie im AWS IAM-Benutzerhandbuch unter Abrufen von Berichten zu Anmeldeinformationen für Ihr AWS Konto.

- Einzelheiten zur API finden Sie unter GenerateCredentialBericht in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Generierung eines neuen Berichts angefordert, was alle vier Stunden erfolgen kann. Wenn der letzte Bericht noch aktuell ist, lautet das Feld Bundesland**COMPLETE**. **Get-IAMCredentialReport**Dient zum Anzeigen des abgeschlossenen Berichts.

```
Request-IAMCredentialReport
```

Ausgabe:

```
Description                                                  State
-----------                                                  -----
No report exists. Starting a new report generation task      STARTED
```

- Einzelheiten zur API finden Sie unter Referenz zum GenerateCredentialMelden von AWS Tools for PowerShell Cmdlets.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response
```

- Einzelheiten zur API finden Sie unter GenerateCredentialReport in AWS SDK for Python (Boto3) API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GenerateServiceLastAccessedDetails`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GenerateServiceLastAccessedDetails`.

CLI

AWS CLI

Beispiel 1: Um einen Servicezugriffsbericht für eine benutzerdefinierte Richtlinie zu generieren

Im folgenden `generate-service-last-accessed-details` Beispiel wird ein Hintergrundjob gestartet, um einen Bericht zu generieren, der die Dienste auflistet, auf die IAM-Benutzer und andere Entitäten zugreifen, mit einer benutzerdefinierten Richtlinie namens`intern-boundary`. Sie können den Bericht anzeigen, nachdem er erstellt wurde, indem Sie den `get-service-last-accessed-details` Befehl ausführen.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Ausgabe:

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

Beispiel 2: Um einen Servicezugriffsbericht für die AWS verwaltete AdministratorAccess Richtlinie zu generieren

Im folgenden `generate-service-last-accessed-details` Beispiel wird ein Hintergrundjob gestartet, um einen Bericht zu generieren, der die Dienste auflistet, auf die IAM-Benutzer und andere Entitäten mit der AWS verwalteten `AdministratorAccess` Richtlinie zugreifen. Sie können den Bericht nach seiner Erstellung anzeigen, indem Sie den `get-service-last-accessed-details` Befehl ausführen.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Ausgabe:

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

Weitere Informationen finden Sie im AWS IAM-Benutzerhandbuch unter Verfeinerung von Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

- Einzelheiten zur API finden Sie unter GenerateServiceLastAccessedDetails in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel entspricht einem API-Cmdlet. GenerateServiceLastAccessedDetails Dies bietet eine Job-ID, die in Get-IAM und ServiceLastAccessedDetail Get-IAM verwendet werden kann ServiceLast AccessedDetail WithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Einzelheiten zur API finden Sie unter GenerateServiceLastAccessedDetails in der Cmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetAccessKeyLastUsed`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetAccessKeyLastUsed`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffsschlüsseln

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);

    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie unter In der AWS SDK for C++ API-Referenz GetAccess KeyLast verwendet.

CLI

AWS CLI

So rufen Sie Informationen darüber ab, wann der angegebene Zugriffsschlüssel zuletzt verwendet wurde

Im folgenden Beispiel werden Informationen darüber abgerufen, wann der Zugriffsschlüssel ABCDEXAMPLE zuletzt verwendet wurde.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Ausgabe:

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Weitere Informationen finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter In der AWS CLI Befehlsreferenz GetAccess KeyLast verwendet.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie den Zugriffsschlüssel ab.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter In der AWS SDK for JavaScript API-Referenz GetAccess KeyLast verwendet.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter In der AWS SDK for JavaScript API-Referenz GetAccess KeyLast verwendet.

PowerShell

Tools für PowerShell

Beispiel 1: Gibt den Namen des Besitzers und Informationen zur letzten Verwendung des angegebenen Zugriffsschlüssels zurück.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Einzelheiten zur API finden Sie unter GetAccessKeyLastUsed in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- Einzelheiten zur API finden Sie unter GetAccessKeyLastUsed in AWS SDK for Python (Boto3) API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetAccountAuthorizationDetails`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetAccountAuthorizationDetails`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

CLI

AWS CLI

Um IAM-Benutzer, -Gruppen, -Rollen und -Richtlinien eines AWS Kontos aufzulisten

Der folgende `get-account-authorization-details` Befehl gibt Informationen zu allen IAM-Benutzern, -Gruppen, -Rollen und -Richtlinien im AWS Konto zurück.

```
aws iam get-account-authorization-details
```

Ausgabe:

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
```

```
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2014-07-30T17:09:20Z",
        "InstanceProfileList": [
            {
                "InstanceProfileId": "AIPA1234567890EXAMPLE",
                "Roles": [
                    {
                        "AssumeRolePolicyDocument": {
                            "Version": "2012-10-17",
                            "Statement": [
                                {
                                    "Sid": "",
                                    "Effect": "Allow",
                                    "Principal": {
                                        "Service": "ec2.amazonaws.com"
                                    },
                                    "Action": "sts:AssumeRole"
                                }
                            ]
                        },
                        "RoleId": "AROA1234567890EXAMPLE",
                        "CreateDate": "2014-07-30T17:09:20Z",
                        "RoleName": "EC2role",
                        "Path": "/",
                        "Arn": "arn:aws:iam::123456789012:role/EC2role"
                    }
                ],
                "CreateDate": "2014-07-30T17:09:20Z",
                "InstanceProfileName": "EC2role",
                "Path": "/",
                "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
            }
        ],
        "RoleName": "EC2role",
        "Path": "/",
        "AttachedManagedPolicies": [
            {
                "PolicyName": "AmazonS3FullAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
            },
```

```
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
```

```
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
```

```
                        "Version": "2012-10-17",
                        "Statement": {
                            "Effect": "Deny",
                            "Action": [
                                "aws-portal:*",
                                "iam:*"
                            ],
                            "Resource": "*"
                        }
                    }
                }
            ],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
        }
    ],
    "Policies": [
        {
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": {
```

```
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
```

```
                                "arn:aws:s3:::example-bucket",
                                "arn:aws:s3:::example-bucket/*"
                            ]
                        }
                    ]
                },
                "IsDefaultVersion": true
            }
        ],
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
        "UpdateDate": "2015-01-21T23:39:41Z"
    },
    {
        "PolicyName": "AmazonEC2FullAccess",
        "CreateDate": "2015-02-06T18:40:15Z",
        "AttachmentCount": 1,
        "IsAttachable": true,
        "PolicyId": "ANPA1234567890EXAMPLE",
        "DefaultVersionId": "v1",
        "PolicyVersionList": [
            {
                "CreateDate": "2014-10-30T20:59:46Z",
                "VersionId": "v1",
                "Document": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Action": "ec2:*",
                            "Effect": "Allow",
                            "Resource": "*"
                        },
                        {
                            "Effect": "Allow",
                            "Action": "elasticloadbalancing:*",
                            "Resource": "*"
                        },
                        {
                            "Effect": "Allow",
                            "Action": "cloudwatch:*",
                            "Resource": "*"
                        },
                        {
```

```
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Weitere Informationen finden Sie unter AWS -Sicherheitsaudit-Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetAccountAuthorizationDetails AWS CLIBefehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Autorisierungsdetails zu den Identitäten im AWS Konto abgerufen und die Elementliste des zurückgegebenen Objekts angezeigt, einschließlich Benutzer, Gruppen und Rollen. In der **`UserDetailList`** Eigenschaft werden beispielsweise Details zu den Benutzern angezeigt. Ähnliche Informationen sind in den **`GroupDetailList`** Eigenschaften **`RoleDetailList`** und verfügbar.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Ausgabe:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
```

```
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Ausgabe:

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Einzelheiten zur API finden Sie unter GetAccountAuthorizationDetails AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Einzelheiten zur API finden Sie GetAccountAuthorizationDetailsin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetAccountPasswordPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetAccountPasswordPolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So zeigen Sie die Passwortrichtlinie für das aktuelle Konto an

Mit dem folgenden `get-account-password-policy`-Befehl werden Details zur Passwortrichtlinie für das aktuelle Konto angezeigt.

```
aws iam get-account-password-policy
```

Ausgabe:

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Wenn keine Passwortwortrichtlinie für das Konto definiert ist, gibt der Befehl einen `NoSuchEntity`-Fehler zurück.

Weitere Informationen finden Sie unter Festlegen einer Kontopasswortrichtlinie für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}
```

```
// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die Passwort-Richtlinie des Kontos ab.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
```

```
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
  console.log(response.PasswordPolicy);
  return response;
};
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details zur Passwortrichtlinie für das aktuelle Konto zurückgegeben. Wenn keine Passwortrichtlinie für das Konto definiert ist, gibt der Befehl einen **`NoSuchEntity`** Fehler zurück.

```
Get-IAMAccountPasswordPolicy
```

Ausgabe:

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Einzelheiten zur API finden Sie unter GetAccountPasswordPolicy AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
 {pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
```

```
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
{pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
{pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
{pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "IAMPolicyManager"
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    begin
      response = @iam_client.get_account_password_policy
      @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
    rescue Aws::IAM::Errors::NoSuchEntity
      @logger.info("The account does not have a password policy.")
    rescue Aws::Errors::ServiceError => e
      @logger.error("Couldn't print the account password policy. Error: #{e.code}
 - #{e.message}")
      raise
    end
  end
end
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn get_account_password_policy(
```

```
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Einzelheiten zur API finden Sie GetAccountPasswordPolicyin der API-Referenz zum AWS SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetAccountSummary`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetAccountSummary`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

CLI

AWS CLI

So rufen Sie Informationen über die Nutzung von IAM-Entitäten und IAM-Kontingenten auf dem aktuellen Konto ab

Der folgende `get-account-summary`-Befehl gibt Informationen zur aktuellen IAM-Entitätsnutzung und zu den aktuellen IAM-Entitätskontingenten im Konto zurück.

```
aws iam get-account-summary
```

Ausgabe:

```
{
    "SummaryMap": {
```

```
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Weitere Informationen zu Entitätsbeschränkungen finden Sie unter IAM- und AWS STS-Kontingente im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetAccountZusammenfassung in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Informationen zur aktuellen Nutzung der IAM-Entität und zu den aktuellen IAM-Entitätskontingenten in der zurückgegeben. AWS-Konto

```
Get-IAMAccountSummary
```

Ausgabe:

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
UsersQuota                              5000
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- Einzelheiten zur API finden Sie unter GetAccountZusammenfassung in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Einzelheiten zur API finden Sie unter GetAccountZusammenfassung in der AWS API-Referenz zum SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetContextKeysForCustomPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetContextKeysForCustomPolicy`.

CLI

AWS CLI

Beispiel 1: Um die Kontextschlüssel aufzulisten, auf die von einer oder mehreren benutzerdefinierten JSON-Richtlinien verwiesen wird, die als Parameter in der Befehlszeile bereitgestellt werden

Der folgende `get-context-keys-for-custom-policy` Befehl analysiert jede bereitgestellte Richtlinie und listet die von diesen Richtlinien verwendeten Kontextschlüssel auf. Verwenden Sie diesen Befehl, um zu ermitteln, welche Kontextschlüsselwerte Sie angeben müssen, um die Befehle `simulate-custom-policy` und `simulate-custom-policy` des Richtliniensimulators erfolgreich verwenden zu können. Mit dem `get-context-keys-for-custom-policy` Befehl können Sie auch die Liste der Kontextschlüssel abrufen, die von allen Richtlinien verwendet werden, die einem IAM-Benutzer oder einer IAM-Rolle zugeordnet sind. Parameterwerte, die mit beginnen, `file://` weisen den Befehl an, die Datei zu lesen und den Inhalt anstelle des Dateinamens selbst als Wert für den Parameter zu verwenden.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

Ausgabe:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Beispiel 2: Um die Kontextschlüssel aufzulisten, auf die von einer oder mehreren benutzerdefinierten JSON-Richtlinien verwiesen wird, die als Dateieingabe bereitgestellt werden

Der folgende `get-context-keys-for-custom-policy` Befehl entspricht dem vorherigen Beispiel, außer dass die Richtlinien in einer Datei und nicht als Parameter bereitgestellt

werden. Da der Befehl eine JSON-Liste von Zeichenfolgen und keine Liste von JSON-Strukturen erwartet, muss die Datei wie folgt strukturiert sein, obwohl Sie sie zu einer zusammenfassen können.

```
[
    "Policy1",
    "Policy2"
]
```

Eine Datei, die die Richtlinie aus dem vorherigen Beispiel enthält, muss also wie folgt aussehen. Sie müssen jedem eingebetteten doppelten Anführungszeichen in der Richtlinienzeichenfolge einen umgekehrten Schrägstrich voranstellen.

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

Diese Datei kann dann an den folgenden Befehl gesendet werden.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

Ausgabe:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Weitere Informationen finden Sie unter Verwenden des IAM-Richtliniensimulators (AWS CLI und AWS API) im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetContextKeysForCustomPolicyin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden alle Kontextschlüssel abgerufen, die in der bereitgestellten Richtlinien-JSON vorhanden sind. Um mehrere Richtlinien bereitzustellen, können Sie sie als kommagetrennte Werteliste angeben.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Einzelheiten zur API finden Sie unter Cmdlet-Referenz. GetContextKeysForCustomPolicyAWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetContextKeysForPrincipalPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetContextKeysForPrincipalPolicy`.

CLI

AWS CLI

Um die Kontextschlüssel aufzulisten, auf die von allen Richtlinien verwiesen wird, die einem IAM-Prinzipal zugeordnet sind

Mit dem folgenden `get-context-keys-for-principal-policy` Befehl werden alle Richtlinien abgerufen, die dem Benutzer `saanvi` und allen Gruppen, denen er angehört, zugeordnet sind. Anschließend analysiert er die einzelnen Richtlinien und listet die von diesen

Richtlinien verwendeten Kontextschlüssel auf. Verwenden Sie diesen Befehl, um zu ermitteln, welche Kontextschlüsselwerte Sie angeben müssen, um die `simulate-principal-policy` Befehle `simulate-custom-policy` und erfolgreich verwenden zu können. Mit dem `get-context-keys-for-custom-policy` Befehl können Sie auch die Liste der Kontextschlüssel abrufen, die von einer beliebigen JSON-Richtlinie verwendet werden.

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

Ausgabe:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Weitere Informationen finden Sie unter Verwenden des IAM-Richtliniensimulators (AWS CLI und AWS API) im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetContextKeysForPrincipalPolicyin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden alle Kontextschlüssel abgerufen, die in der bereitgestellten Policy-JSON-Datei und in den Richtlinien enthalten sind, die der IAM-Entität zugeordnet sind (Benutzer/Rolle usw.). Für — PolicyInputList Sie können eine Liste mit mehreren Werten als durch Kommas getrennte Werte angeben.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
```

```
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Einzelheiten zur API finden Sie unter GetContextKeysForPrincipalPolicy AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetCredentialReport`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetCredentialReport`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

CLI

AWS CLI

So rufen Sie einen Bericht zu Anmeldeinformationen ab

In diesem Beispiel wird der zurückgegebene Bericht geöffnet und als Array von Textzeilen an die Pipeline ausgegeben.

```
aws iam get-credential-report
```

Ausgabe:

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

Weitere Informationen finden Sie im AWS IAM-Benutzerhandbuch unter Abrufen von Berichten zu Anmeldeinformationen für Ihr AWS Konto.

- Einzelheiten zur API finden Sie unter GetCredentialBericht in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der zurückgegebene Bericht geöffnet und als Array von Textzeilen an die Pipeline ausgegeben. Die erste Zeile ist die Kopfzeile mit durch Kommas getrennten Spaltennamen. Jede nachfolgende Zeile ist die Detailzeile für einen Benutzer, wobei jedes Feld durch Kommas getrennt ist. Bevor Sie den Bericht anzeigen können, müssen Sie ihn mit dem **`Request-IAMCredentialReport`** Cmdlet generieren. Um den Bericht als einzelne Zeichenfolge abzurufen, verwenden Sie **`-Raw`** anstelle von. **`-AsTextArray`** Der Alias **`-SplitLines`** wird auch für den **`-AsTextArray`** Switch akzeptiert. Die vollständige Liste der Spalten in der Ausgabe finden Sie in der Service-API-Referenz. Beachten Sie, dass Sie, wenn Sie **`-AsTextArray`** oder nicht verwenden**`-SplitLines`**, den Text mithilfe der **`StreamReader`** .NET-Klasse aus der **`.Content`** Eigenschaft extrahieren müssen.

```
Request-IAMCredentialReport
```

Ausgabe:

```
Description                                                        State
-----------                                                        -----
No report exists. Starting a new report generation task            STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Ausgabe:

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Einzelheiten zur API finden Sie unter GetCredentialReport in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Einzelheiten zur API finden Sie unter GetCredentialReport in AWS SDK for Python (Boto3) API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetGroup`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetGroup`.

CLI

AWS CLI

Um eine IAM-Gruppe zu erhalten

In diesem Beispiel werden Details zur IAM-Gruppe zurückgegeben. `Admins`

```
aws iam get-group \
    --group-name Admins
```

Ausgabe:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
}
```

Weitere Informationen finden Sie unter IAM-Identitäten (Benutzer, Benutzergruppen und Rollen) im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetGroupin AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details zur IAM-Gruppe zurückgegeben**`Testers`**, einschließlich einer Sammlung aller IAM-Benutzer, die zu der Gruppe gehören.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Ausgabe:

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Ausgabe:

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Einzelheiten zur API finden Sie unter GetGroup AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetGroupPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetGroupPolicy`.

CLI

AWS CLI

Um Informationen über eine Richtlinie abzurufen, die einer IAM-Gruppe zugeordnet ist

Mit dem folgenden `get-group-policy` Befehl werden Informationen über die angegebene Richtlinie abgerufen, die der genannten `Test-Group` Gruppe zugeordnet ist.

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

Ausgabe:

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetGroupRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details zur eingebetteten Inline-Richtlinie zurückgegeben, die **`PowerUserAccess-Testers`** nach der Gruppe benannt ist**`Testers`**. Die **`PolicyDocument`** Eigenschaft ist URL-codiert. Sie wird in diesem Beispiel mit der **`UrlDecode`** .NET-Methode dekodiert.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Ausgabe:

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "NotAction": "iam:*",
      "Resource": "*"
    }
  ]
}
```

- Einzelheiten zur API finden Sie unter GetGroupRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetInstanceProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetInstanceProfile`.

CLI

AWS CLI

Um Informationen über ein Instanzprofil zu erhalten

Der folgende `get-instance-profile` Befehl ruft Informationen über das angegebene Instanzprofil ab`ExampleInstanceProfile`.

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Ausgabe:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
    }
}
```

Weitere Informationen finden Sie unter Verwenden von Instance-Profilen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetInstanceProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details des genannten Instanzprofils zurückgegeben**ec2instancerole**, das im aktuellen AWS Konto definiert ist.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Ausgabe:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Einzelheiten zur API finden Sie unter GetInstanceProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **GetLoginProfile** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetLoginProfile`.

CLI

AWS CLI

Um Passwortinformationen für einen IAM-Benutzer abzurufen

Mit dem folgenden `get-login-profile` Befehl werden Informationen zum Passwort für den IAM-Benutzer mit dem Namen abgerufen. Bob

```
aws iam get-login-profile \
    --user-name Bob
```

Ausgabe:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

Der `get-login-profile` Befehl kann verwendet werden, um zu überprüfen, ob ein IAM-Benutzer ein Passwort hat. Der Befehl gibt einen `NoSuchEntity` Fehler zurück, wenn kein Passwort für den Benutzer definiert ist.

Mit diesem Befehl können Sie kein Passwort anzeigen. Wenn das Passwort verloren gegangen ist, können Sie das Passwort (`update-login-profile`) für den Benutzer zurücksetzen. Alternativ können Sie das Anmeldeprofil (`delete-login-profile`) für den Benutzer löschen und dann ein neues erstellen (`create-login-profile`).

Weitere Informationen finden Sie unter Passwörter für IAM-Benutzer verwalten im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetLoginProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel gibt das Erstellungsdatum des Kennworts zurück und gibt an, ob für den IAM-Benutzer **`David`** ein Zurücksetzen des Kennworts erforderlich ist.

```
Get-IAMLoginProfile -UserName David
```

Ausgabe:

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
```

```
12/10/2014 3:39:44 PM        False                                  David
```

- Einzelheiten zur API finden Sie unter GetLoginProfile in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetOpenIdConnectProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetOpenIdConnectProvider`.

CLI

AWS CLI

Um Informationen über den angegebenen OpenID Connect-Anbieter zurückzugeben

In diesem Beispiel werden Details über den OpenID Connect-Anbieter zurückgegeben, dessen ARN lautet`arn:aws:iam::123456789012:oidc-provider/server.example.com`.

```
aws iam get-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Ausgabe:

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

Weitere Informationen finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetOpenIdConnectAnbieter in der Befehlsreferenz.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel gibt Details über den OpenID Connect-Anbieter zurück, dessen ARN lautet**`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`**. Die **`ClientIDList`** Eigenschaft ist eine Sammlung, die alle für diesen Anbieter definierten Client-IDs enthält.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Ausgabe:

```
ClientIDList         CreateDate               ThumbprintList
           Url
------------         ----------               --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Einzelheiten zur API finden Sie unter GetOpenIdConnectAnbieter in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Arbeiten mit der IAM-Policy-Builder-API

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }
```

- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
```

```
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So rufen Sie Informationen über die angegebene verwaltete Richtlinie ab

In diesem Beispiel werden Details über die verwaltete Richtlinie mit dem ARN `arn:aws:iam::123456789012:policy/MySamplePolicy` zurückgegeben.

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Ausgabe:

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetPolicyin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
```

```
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die Richtlinie ab.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });
```

```
  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Einzelheiten zur API finden Sie GetPolicyin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
```

```
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details zu der verwalteten Richtlinie zurückgegeben, deren ARN lautet**`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Ausgabe:

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Einzelheiten zur API finden Sie unter GetPolicy AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Einzelheiten zur API finden Sie GetPolicyin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
```

```
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie GetPolicyin der AWS SDK for Ruby API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
        do {
            let output = try await client.getPolicy(input: input)
            guard let policy = output.policy else {
```

```
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie GetPolicyin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetPolicyVersion`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetPolicyVersion`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Verwalten von Richtlinien
- Arbeiten mit der IAM-Policy-Builder-API

CLI

AWS CLI

So rufen Sie Informationen über die angegebene Version der angegebenen verwalteten Richtlinie ab

In diesem Beispiel wird das Richtliniendokument für die Version v2 der Richtlinie zurückgegeben, deren ARN `arn:aws:iam::123456789012:policy/MyManagedPolicy` lautet.

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Ausgabe:

```
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetPolicyVersion in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Richtliniendokument für die **v2** Version der Richtlinie zurückgegeben, deren ARN lautet**arn:aws:iam::123456789012:policy/MyManagedPolicy**. Das Richtliniendokument in der **Document** Eigenschaft ist URL-kodiert und wird in diesem Beispiel mit der **UrlDecode** .NET-Methode dekodiert.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Ausgabe:

```
CreateDate             Document
 IsDefaultVersion     VersionId
```

```
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version": "2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
}
```

- Einzelheiten zur API finden Sie unter GetPolicyVersion in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
```

```
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
policy_arn)
        raise
    else:
        return policy_statement
```

- Einzelheiten zur API finden Sie unter GetPolicyVersion in der AWS API-Referenz zum SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
```

```
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }
```

- Einzelheiten zur API finden Sie GetRolein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So rufen Sie Informationen über eine IAM-Rolle ab

Mit dem folgenden `get-role`-Befehl werden Informationen über die Rolle mit dem Namen `Test-Role` abgerufen.

```
aws iam get-role \
    --role-name Test-Role
```

Ausgabe:

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
```

```
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

Der Befehl zeigt die Vertrauensrichtlinie an, die der Rolle zugeordnet ist. Verwenden Sie den `list-role-policies`-Befehl, um die einer Rolle zugeordneten Berechtigungsrichtlinien aufzulisten.

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetRolein der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
```

```
  role = result.Role
 }
 return role, err
}
```

- Einzelheiten zur API finden Sie GetRolein der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die Rolle ab.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetRolein der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Einzelheiten zur API finden Sie GetRolein der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel gibt die Details von zurück**`lamda_exec_role`**. Es enthält das Dokument mit der Vertrauensrichtlinie, in dem angegeben ist, wer diese Rolle übernehmen kann. Das Richtliniendokument ist URL-kodiert und kann mit der **`UrlDecode`** .NET-Methode dekodiert werden. In diesem Beispiel wurden bei der ursprünglichen Richtlinie alle Leerzeichen entfernt, bevor sie in die Richtlinie hochgeladen wurde. Um die Dokumente mit den Berechtigungsrichtlinien einzusehen, in denen festgelegt ist, was jemand, der die Rolle übernimmt, tun kann, verwenden Sie die **`Get-IAMRolePolicy`** Option für Inline-Richtlinien und **`Get-IAMPolicyVersion`** für angehängte verwaltete Richtlinien.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Ausgabe:

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Ausgabe:

```
{"Version":"2012-10-17","Statement":[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Einzelheiten zur API finden Sie unter GetRole AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
```

```
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Einzelheiten zur API finden Sie GetRolein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
                                role_name: name,
                              }).role
  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
  role
end
```

- Einzelheiten zur API finden Sie GetRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Einzelheiten zur API finden Sie GetRolein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func getRole(name: String) async throws -> IAMClientTypes.Role {
    let input = GetRoleInput(
        roleName: name
    )
    do {
        let output = try await client.getRole(input: input)
        guard let role = output.role else {
            throw ServiceHandlerError.noSuchRole
        }
        return role
    } catch {
        throw error
    }
}
```

- Einzelheiten zur API finden Sie GetRolein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetRolePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetRolePolicy`.

CLI

AWS CLI

Um Informationen über eine Richtlinie abzurufen, die einer IAM-Rolle zugeordnet ist

Mit dem folgenden `get-role-policy` Befehl werden Informationen über die angegebene Richtlinie abgerufen, die der genannten `Test-Role` Rolle zugeordnet ist.

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Ausgabe:

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetRoleRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Dokument mit der Berechtigungsrichtlinie für die angegebene Richtlinie zurückgegeben**`oneClick_lambda_exec_role_policy`**, die in die IAM-Rolle **`lamda_exec_role`** eingebettet ist. Das resultierende Richtliniendokument ist URL-kodiert. In diesem Beispiel wird es mit der **`UrlDecode`** .NET-Methode dekodiert.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Ausgabe:

```
PolicyDocument                                                  PolicyName
              UserName
--------------                                                  ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Ausgabe:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    }
  ]
}
```

- Einzelheiten zur API finden Sie unter GetRoleRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetSamlProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetSamlProvider`.

CLI

AWS CLI

Um das SAML-Provider-Metadokument abzurufen

In diesem Beispiel werden die Details über den SAML 2.0-Anbieter abgerufen, dessen ARM ist. `arn:aws:iam::123456789012:saml-provider/SAMLADFS` Die Antwort enthält das Metadatendokument, das Sie vom Identitätsanbieter zur Erstellung der AWS SAML-Provider-Entität erhalten haben, sowie die Erstellungs- und Ablaufdaten.

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Ausgabe:

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
```

```
    ]
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-SAML-Identitätsanbietern im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetSamlAnbieter in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden die Details zum SAML 2.0-Anbieter abgerufen, dessen ARM arn:aws:iam: :123456789012:SAML-Provider/samladfs ist. Die Antwort enthält das Metadatendokument, das Sie vom Identitätsanbieter zur Erstellung der SAML-Provider-Entität erhalten haben, sowie die Erstellungs- und Ablaufdaten. AWS

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Ausgabe:

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Einzelheiten zur API finden Sie unter GetSamlAnbieter in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetServerCertificate`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetServerCertificate`.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }
```

```
    return result;
}
```

- Einzelheiten zur API finden Sie unter GetServerZertifikat in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um Details zu einem Serverzertifikat in Ihrem AWS Konto zu erhalten

Mit dem folgenden `get-server-certificate` Befehl werden alle Details zum angegebenen Serverzertifikat in Ihrem AWS Konto abgerufen.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Ausgabe:

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
```

```
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

Verwenden Sie den `list-server-certificates` Befehl, um die in Ihrem AWS Konto verfügbaren Serverzertifikate aufzulisten.

Weitere Informationen finden Sie unter Verwaltung von Serverzertifikaten in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetServerZertifikat in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie ein Serverzertifikat ab.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter GetServerZertifikat in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getServerCertificate(
```

```
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter GetServerZertifikat in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details über das angegebene Serverzertifikat abgerufen. **`MyServerCertificate`** Sie finden die Zertifikatsdetails in den **`ServerCertificateMetadata`** Eigenschaften **`CertificateBody`** und.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Ausgabe:

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
```

```
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Ausgabe:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Einzelheiten zur API finden Sie unter GetServerCertificate in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetServiceLastAccessedDetails`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetServiceLastAccessedDetails`.

CLI

AWS CLI

Um einen Servicezugriffsbericht abzurufen

Im folgenden `get-service-last-accessed-details` Beispiel wird ein zuvor generierter Bericht abgerufen, der die Dienste auflistet, auf die IAM-Entitäten zugreifen. Verwenden Sie den Befehl, um einen Bericht zu generieren. `generate-service-last-accessed-details`

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Ausgabe:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

Weitere Informationen finden Sie im AWS IAM-Benutzerhandbuch unter Verfeinerung von Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

- Einzelheiten zur API finden Sie unter GetServiceLastAccessedDetails in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel enthält Details zum Dienst, auf den die IAM-Entität (Benutzer, Gruppe, Rolle oder Richtlinie) zuletzt zugegriffen hat, die dem Anforderungsaufruf zugeordnet ist.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Ausgabe:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Einzelheiten zur API finden Sie unter GetServiceLastAccessedDetails in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetServiceLastAccessedDetailsWithEntities`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetServiceLastAccessedDetailsWithEntities`.

CLI

AWS CLI

Um einen Servicezugriffsbericht mit Details für einen Dienst abzurufen

Im folgenden `get-service-last-accessed-details-with-entities` Beispiel wird ein Bericht abgerufen, der Details zu IAM-Benutzern und anderen Entitäten enthält, die auf den angegebenen Dienst zugegriffen haben. Verwenden Sie den Befehl, um einen Bericht zu

generieren. `generate-service-last-accessed-details` Um eine Liste der Dienste abzurufen, auf die über Namespaces zugegriffen wird, verwenden Sie. `get-service-last-accessed-details`

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

Ausgabe:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Verfeinerung von Berechtigungen bei der AWS Verwendung von Informationen, auf die AWS zuletzt zugegriffen wurde.

- Einzelheiten zur API finden Sie unter GetServiceLastAccessedDetailsWithEntitäten in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel liefert den Zeitstempel, auf den zuletzt zugegriffen wurde, für den Dienst in der Anfrage der jeweiligen IAM-Entität.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Ausgabe:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Ausgabe:

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Ausgabe:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
```

```
Type : USER
```

- Einzelheiten zur API finden Sie unter GetServiceLastAccessedDetailsWithEntitäten in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetServiceLinkedRoleDeletionStatus`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetServiceLinkedRoleDeletionStatus`.

CLI

AWS CLI

So überprüfen Sie den Status einer Anfrage zum Löschen einer serviceverknüpften Rolle

Im folgenden `get-service-linked-role-deletion-status`-Beispiel wird der Status einer früheren Anfrage zum Löschen einer serviceverknüpften Rolle angezeigt. Der Löschvorgang erfolgt asynchron. Wenn Sie die Anfrage stellen, erhalten Sie einen `DeletionTaskId`-Wert, den Sie als Parameter für diesen Befehl angeben.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Ausgabe:

```
{
"Status": "SUCCEEDED"
}
```

Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetServiceLinkedRoleDeletionStatusin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetServiceLinkedRoleDeletionStatusin der AWS SDK for JavaScript API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetUser`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Einzelheiten zur API finden Sie GetUserin der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Einzelheiten zur API finden Sie GetUserin der AWS CLI Befehlsreferenz.

CLI

AWS CLI

So rufen Sie Informationen über einen IAM-Benutzer ab

Mit dem folgenden `get-user`-Befehl werden Informationen über den IAM-Benutzer mit dem Namen `Paulo` abgerufen.

```
aws iam get-user \
    --user-name Paulo
```

Ausgabe:

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

Weitere Informationen finden Sie unter Verwalten von IAM-Benutzern im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie GetUserin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

- Einzelheiten zur API finden Sie GetUserin der AWS SDK for Go API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details über den genannten **David** Benutzer abgerufen.

```
Get-IAMUser -UserName David
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Beispiel 2: In diesem Beispiel werden Details über den aktuell angemeldeten IAM-Benutzer abgerufen.

```
Get-IAMUser
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Einzelheiten zur API finden Sie unter Cmdlet-Referenz. GetUserAWS Tools for PowerShell

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Retrieves a user's details
#
```

```
  # @param user_name [String] The name of the user to retrieve
  # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
 error occurred
  def get_user(user_name)
    response = @iam_client.get_user(user_name: user_name)
    response.user
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("User '#{user_name}' not found.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error retrieving user '#{user_name}': #{e.message}")
    nil
  end
```

- Einzelheiten zur API finden Sie GetUserin der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetUserPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetUserPolicy`.

CLI

AWS CLI

Um Richtliniendetails für einen IAM-Benutzer aufzulisten

Der folgende `get-user-policy` Befehl listet die Details der angegebenen Richtlinie auf, die dem genannten IAM-Benutzer zugeordnet ist. Bob

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Ausgabe:

```
{
    "UserName": "Bob",
```

```
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
}
```

Verwenden Sie den `list-user-policies`-Befehl, um eine Liste der Richtlinien für einen IAM-Benutzer abzurufen.

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetUserRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden die Details der genannten Inline-Richtlinie abgerufen**`Davids_IAM_Admin_Policy`**, die in den IAM-Benutzer mit dem Namen eingebettet ist. **`David`** Das Richtliniendokument ist URL-codiert.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Ausgabe:

```
PolicyDocument                                          PolicyName
       UserName
--------------                                          ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
```

```
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

- Einzelheiten zur API finden Sie unter GetUserRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListAccessKeys`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListAccessKeys`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffsschlüsseln

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
```

```
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
    --output text \
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
```

```
                        std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                        statusString << std::setw(20) <<
                        key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So listen Sie die Zugriffsschlüssel-IDs für einen IAM-Benutzer auf

Der folgende `list-access-keys`-Befehl listet die Zugriffsschlüssel-IDs für den IAM-Benutzer mit dem Namen Bob auf.

```
aws iam list-access-keys \
    --user-name Bob
```

Ausgabe:

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
```

```
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

Sie können die geheimen Zugriffsschlüssel für IAM-Benutzer nicht auflisten. Bei Verlust der geheimen Zugangsschlüssel müssen Sie mit dem `create-access-keys`-Befehl neue Zugangsschlüssel erstellen.

Weitere Informationen finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
```

```
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user for which access keys are
 retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
 ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();
```

```
                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
 ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
                            .build();

                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
 metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Zugriffsschlüssel auf.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
```

```
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
```

```
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
            userName = userNameVal
        }
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- API-Details finden Sie unter ListAccessKeys in AWS SDK for Kotlin API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl listet die Zugriffsschlüssel für den IAM-Benutzer mit dem Namen **Bob** auf. Beachten Sie, dass Sie die geheimen Zugriffsschlüssel für IAM-Benutzer nicht auflisten

können. Wenn die geheimen Zugriffsschlüssel verloren gehen, müssen Sie mit dem **New-IAMAccessKey** Cmdlet neue Zugriffsschlüssel erstellen.

```
Get-IAMAccessKey -UserName "Bob"
```

Ausgabe:

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Einzelheiten zur API finden Sie unter Referenz zu ListAccessSchlüsseln im AWS Tools for PowerShell Cmdlet.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
```

```
    else:
        return keys
```

- API-Einzelheiten finden Sie unter ListAccessKeys in AWS SDK for Python (Boto3) API-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Dieses Beispielmodul listet Zugriffsschlüssel auf, erstellt, deaktiviert und löscht sie.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
```

```
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
```

```
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter ListAccessSchlüssel in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListAccountAliases`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListAccountAliases`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten Ihrer Konten

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So listen Sie Konto-Aliase auf

Der folgende `list-account-aliases`-Befehl listet die Aliase für das aktuelle Konto auf.

```
aws iam list-account-aliases
```

Ausgabe:

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

Weitere Informationen finden Sie unter Ihre AWS Konto-ID und deren Alias im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccountAliases {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Konto-Aliase auf.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
```

```
}
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listAliases() {
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- API-Details finden Sie unter ListAccountAliase im AWS SDK für die Kotlin-API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl gibt den Kontoalias für den zurück AWS-Konto.

```
Get-IAMAccountAlias
```

Ausgabe:

```
ExampleCo
```

- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS API-Referenz zum SDK for Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Kontenaliase auflisten, erstellen und löschen.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter ListAccountAliase in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListAttachedGroupPolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListAttachedGroupPolicies`.

CLI

AWS CLI

Um alle verwalteten Richtlinien aufzulisten, die der angegebenen Gruppe zugeordnet sind

In diesem Beispiel werden die Namen und ARNs der verwalteten Richtlinien zurückgegeben, die der `Admins` im Konto genannten IAM-Gruppe zugeordnet sind. AWS

```
aws iam list-attached-group-policies \
    --group-name Admins
```

Ausgabe:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListAttachedGroupPolicies AWS CLIBefehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl gibt die Namen und ARNs der verwalteten Richtlinien zurück, die der **`Admins`** im Konto genannten IAM-Gruppe zugeordnet sind. AWS Verwenden Sie den Befehl, um die Liste der in die Gruppe eingebetteten Inline-Richtlinien anzuzeigen. **`Get-IAMGroupPolicyList`**

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Ausgabe:

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/SecurityAudit                    SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess              AdministratorAccess
```

- Einzelheiten zur API finden Sie unter ListAttachedGroupPolicies AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListAttachedRolePolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListAttachedRolePolicies`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   /// <summary>
   /// List the IAM role policies that are attached to an IAM role.
   /// </summary>
   /// <param name="roleName">The IAM role to list IAM policies for.</param>
   /// <returns>A list of the IAM policies attached to the IAM role.</returns>
   public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
   {
       var attachedPolicies = new List<AttachedPolicyType>();
       var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

       await foreach (var response in attachedRolePoliciesPaginator.Responses)
       {
           attachedPolicies.AddRange(response.AttachedPolicies);
       }

       return attachedPolicies;
   }
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So listen Sie alle verwalteten Richtlinien auf, die der angegebenen Rolle angefügt sind

Dieser Befehl gibt die Namen und ARNs der verwalteten Richtlinien zurück, die der `SecurityAuditRole` im Konto genannten IAM-Rolle zugeordnet sind. AWS

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Ausgabe:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Richtlinien auf, die an eine Rolle angefügt sind.

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });

  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl gibt die Namen und ARNs der verwalteten Richtlinien zurück, die der **`SecurityAuditRole`** im Konto genannten IAM-Rolle zugeordnet sind. AWS Verwenden Sie den Befehl, um die Liste der Inline-Richtlinien anzuzeigen, die in die Rolle eingebettet sind. **`Get-IAMRolePolicyList`**

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Ausgabe:

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/SecurityAudit                    SecurityAudit
```

- Einzelheiten zur API finden Sie unter ListAttachedRolePolicies AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
```

```
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispielmodul werden Rollenrichtlinien aufgelistet, erstellt, angehängt und entfernt.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
```

```
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
```

```
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter ListAttachedRolePolicies AWS SDK for RubyAPI-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_attached_role_policies(
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
[IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListAttachedRolePoliciesInput(
                marker: marker,
                roleName: role
            )
            let output = try await client.listAttachedRolePolicies(input: input)

            guard let attachedPolicies = output.attachedPolicies else {
                return policyList
            }

            for attachedPolicy in attachedPolicies {
                policyList.append(attachedPolicy)
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Einzelheiten zur API finden Sie ListAttachedRolePoliciesin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListAttachedUserPolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListAttachedUserPolicies`.

CLI

AWS CLI

Um alle verwalteten Richtlinien aufzulisten, die dem angegebenen Benutzer zugeordnet sind

Dieser Befehl gibt die Namen und ARNs der verwalteten Richtlinien für den IAM-Benutzer zurück, der Bob AWS im Konto angegeben ist.

```
aws iam list-attached-user-policies \
    --user-name Bob
```

Ausgabe:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListAttachedUserPoliciesin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieser Befehl gibt die Namen und ARNs der verwalteten Richtlinien für den IAM-Benutzer zurück, der **Bob** AWS im Konto angegeben ist. Verwenden Sie den Befehl, um die Liste der Inline-Richtlinien anzuzeigen, die in den IAM-Benutzer eingebettet sind. **Get-IAMUserPolicyList**

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Ausgabe:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/TesterPolicy                      TesterPolicy
```

- Einzelheiten zur API finden Sie unter ListAttachedUserPolicies AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **ListEntitiesForPolicy** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListEntitiesForPolicy`.

CLI

AWS CLI

Um alle Benutzer, Gruppen und Rollen aufzulisten, denen die angegebene verwaltete Richtlinie zugeordnet ist

In diesem Beispiel wird eine Liste von IAM-Gruppen, -Rollen und Benutzern zurückgegeben, denen die Richtlinie `arn:aws:iam::123456789012:policy/TestPolicy` angehängt ist.

```
aws iam list-entities-for-policy \
```

```
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Ausgabe:

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListEntitiesForPolicy AWS CLIBefehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Liste von IAM-Gruppen, -Rollen und Benutzern zurückgegeben, denen die Richtlinie **`arn:aws:iam::123456789012:policy/TestPolicy`** zugewiesen wurde.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Ausgabe:

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Einzelheiten zur API finden Sie unter ListEntitiesForPolicy AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListGroupPolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListGroupPolicies`.

CLI

AWS CLI

Um alle Inline-Richtlinien aufzulisten, die der angegebenen Gruppe zugeordnet sind

Der folgende `list-group-policies` Befehl listet die Namen der Inline-Richtlinien auf, die an die `Admins` im aktuellen Konto angegebene IAM-Gruppe angehängt sind.

```
aws iam list-group-policies \
    --group-name Admins
```

Ausgabe:

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
}
```

Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListGroupRichtlinien in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Liste der Inline-Richtlinien zurückgegeben, die in die Gruppe eingebettet sind**`Testers`**. Verwenden Sie den Befehl, um die verwalteten Richtlinien abzurufen, die der Gruppe zugeordnet sind**`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Ausgabe:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Einzelheiten zur API finden Sie unter ListGroupRichtlinien in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListGroups`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListGroups`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
```

```
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
 ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Einzelheiten zur API finden Sie ListGroupsin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So listen Sie die IAM-Gruppen für das aktuelle Konto auf

Der folgende `list-groups`-Befehl listet die IAM-Gruppen im aktuellen Konto auf.

```
aws iam list-groups
```

Ausgabe:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
```

```
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwaltung von IAM-Benutzergruppen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListGroupsin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(maxGroups int32) ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(context.TODO(),
 &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
```

```
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
 return groups, err
}
```

- Einzelheiten zur API finden Sie ListGroupsin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Gruppen auf.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);
```

```
  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Einzelheiten zur API finden Sie ListGroupsin der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
```

```
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- Einzelheiten zur API finden Sie ListGroupsin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel gibt eine Sammlung aller in der aktuellen AWS-Konto Version definierten IAM-Gruppen zurück.

```
Get-IAMGroupList
```

Ausgabe:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Einzelheiten zur API finden Sie unter ListGroups AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_groups(count):
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Einzelheiten zur API finden Sie ListGroupsin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
```

```
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Einzelheiten zur API finden Sie ListGroupsin der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
```

```
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Einzelheiten zur API finden Sie ListGroupsin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func listGroups() async throws -> [String] {
        var groupList: [String] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListGroupsInput(marker: marker)
            let output = try await client.listGroups(input: input)

            guard let groups = output.groups else {
                return groupList
```

```
            }

            for group in groups {
                if let name = group.groupName {
                    groupList.append(name)
                }
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return groupList
    }
```

- Einzelheiten zur API finden Sie ListGroupsin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListGroupsForUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListGroupsForUser`.

CLI

AWS CLI

Um die Gruppen aufzulisten, zu denen ein IAM-Benutzer gehört

Der folgende `list-groups-for-user` Befehl zeigt die Gruppen an, zu denen der angegebene IAM-Benutzer Bob gehört.

```
aws iam list-groups-for-user \
    --user-name Bob
```

Ausgabe:

```
{
    "Groups": [
        {
            "Path": "/",
```

```
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwaltung von IAM-Benutzergruppen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListGroupsForUserin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Liste der IAM-Gruppen zurückgegeben, zu denen der IAM-Benutzer **`David`** gehört.

```
Get-IAMGroupForUser -UserName David
```

Ausgabe:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /
```

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Einzelheiten zur API finden Sie unter ListGroupsForUser AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListInstanceProfiles`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListInstanceProfiles`.

CLI

AWS CLI

Um die Instanzprofile für das Konto aufzulisten

Der folgende `list-instance-profiles` Befehl listet die Instanzprofile auf, die dem aktuellen Konto zugeordnet sind.

```
aws iam list-instance-profiles
```

Ausgabe:

```
{
    "InstanceProfiles": [
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
```

```
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
```

```
                }
            ]
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwenden von Instance-Profilen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListInstanceProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel gibt eine Sammlung der Instanzprofile zurück, die in der aktuellen Version definiert sind AWS-Konto.

```
Get-IAMInstanceProfileList
```

Ausgabe:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Einzelheiten zur API finden Sie unter Referenz zu ListInstanceProfilen in AWS Tools for PowerShell Cmdlets.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListInstanceProfilesForRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListInstanceProfilesForRole`.

CLI

AWS CLI

Um die Instanzprofile für eine IAM-Rolle aufzulisten

Der folgende `list-instance-profiles-for-role` Befehl listet die Instanzprofile auf, die der Rolle `Test-Role` zugeordnet sind.

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

Ausgabe:

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwenden von Instance-Profilen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListInstanceProfilesForRolle in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details des mit der Rolle verknüpften Instanzprofils zurückgegeben**ec2instancerole**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Ausgabe:

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Einzelheiten zur API finden Sie unter Referenz zur ListInstanceProfilesForRolle im AWS Tools for PowerShell Cmdlet.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **ListMfaDevices** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListMfaDevices`.

CLI

AWS CLI

Um alle MFA-Geräte für einen bestimmten Benutzer aufzulisten

In diesem Beispiel werden Details über das MFA-Gerät zurückgegeben, das dem IAM-Benutzer zugewiesen wurde. `Bob`

```
aws iam list-mfa-devices \
    --user-name Bob
```

Ausgabe:

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListMfaGeräte in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details über das MFA-Gerät zurückgegeben, das dem IAM-Benutzer zugewiesen wurde. **`David`** In diesem Beispiel können Sie erkennen, dass es sich um ein virtuelles Gerät **`SerialNumber`** handelt, da es sich um einen ARN und nicht um die tatsächliche Seriennummer eines physischen Geräts handelt.

```
Get-IAMMFADevice -UserName David
```

Ausgabe:

```
EnableDate                SerialNumber                           UserName
----------                ------------                           --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David    David
```

- Einzelheiten zur API finden Sie unter ListMfaDevices in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListOpenIdConnectProviders`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListOpenIdConnectProviders`.

CLI

AWS CLI

Um Informationen über die OpenID Connect-Anbieter im AWS Konto aufzulisten

Dieses Beispiel gibt eine Liste der ARNS aller OpenID Connect-Anbieter zurück, die im AWS Girokonto definiert sind.

```
aws iam list-open-id-connect-providers
```

Ausgabe:

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
```

```
}
```

Weitere Informationen finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListOpenIdConnectAnbieter in der Befehlsreferenz.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel gibt eine Liste von ARNS aller OpenID Connect-Anbieter zurück, die in der aktuellen Version definiert sind. AWS-Konto

```
Get-IAMOpenIDConnectProviderList
```

Ausgabe:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Einzelheiten zur API finden Sie unter ListOpenIdConnectProviders in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListPolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListPolicies`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Richtlinien

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }
```

```
    return true;
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um verwaltete Richtlinien aufzulisten, die für Ihr AWS Konto verfügbar sind

In diesem Beispiel wird eine Sammlung der ersten beiden verwalteten Richtlinien zurückgegeben, die im aktuellen AWS Konto verfügbar sind.

```
aws iam list-policies \
    --max-items 3
```

Ausgabe:

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
```

```
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListPoliciesunter AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
```

```
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listet Sie die Richtlinien auf.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
```

```
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Einzelheiten zur API finden Sie ListPoliciesin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Sammlung der ersten drei verwalteten Richtlinien zurückgegeben, die im aktuellen AWS Konto verfügbar sind. Da nicht angegeben **`-scope`** ist, werden standardmäßig sowohl AWS verwaltete als auch vom **`all`** Kunden verwaltete Richtlinien verwendet und diese sind auch enthalten.

```
Get-IAMPolicyList -MaxItem 3
```

Ausgabe:

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Beispiel 2: In diesem Beispiel wird eine Sammlung der ersten beiden vom Kunden verwalteten Policen zurückgegeben, die im AWS Girokonto verfügbar sind. Es wird verwendet **`-Scope local`**, um die Ausgabe nur auf vom Kunden verwaltete Policen zu beschränken.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Einzelheiten zur API finden Sie unter ListPolicies AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_policies(scope):
```

```
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Einzelheiten zur API finden Sie ListPoliciesin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispielmodul werden Rollenrichtlinien aufgelistet, erstellt, angehängt und entfernt.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
```

```
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
```

```
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter ListPolicies AWS SDK for RubyAPI-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Einzelheiten zur API finden Sie ListPoliciesin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func listPolicies() async throws -> [MyPolicyRecord] {
    var policyList: [MyPolicyRecord] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListPoliciesInput(marker: marker)
        let output = try await client.listPolicies(input: input)

        guard let policies = output.policies else {
            return policyList
        }

        for policy in policies {
            guard   let name = policy.policyName,
                    let id = policy.policyId,
                    let arn = policy.arn else {
                throw ServiceHandlerError.noSuchPolicy
            }
            policyList.append(MyPolicyRecord(name: name, id: id, arn: arn))
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return policyList
```

```
    }
```

- Einzelheiten zur API finden Sie ListPoliciesin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListPolicyVersions`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListPolicyVersions`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Verwalten von Richtlinien
- Zurücksetzen einer Richtlinienversion

CLI

AWS CLI

Um Informationen zu den Versionen der angegebenen verwalteten Richtlinie aufzulisten

In diesem Beispiel wird die Liste der verfügbaren Versionen der Richtlinie zurückgegeben, deren ARN lautet`arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Ausgabe:

```
{
    "IsTruncated": false,
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
```

```
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListPolicyVersionen in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Liste der verfügbaren Versionen der Richtlinie zurückgegeben, deren ARN lautet**`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Um das Richtliniendokument für eine bestimmte Version abzurufen, verwenden Sie den **`Get-IAMPolicyVersion`** Befehl und geben Sie die gewünschte Version an. **`VersionId`**

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Ausgabe:

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- Einzelheiten zur API finden Sie unter ListPolicyVersionen in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListRolePolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListRolePolicies`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So listen Sie die einer IAM-Rolle angefügten Richtlinien auf

Mit dem folgenden `list-role-policies`-Befehl werden die Namen der Berechtigungsrichtlinien für die angegebene IAM-Rolle aufgelistet.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Ausgabe:

```
{
    "PolicyNames": [
        "ExamplePolicy"
    ]
}
```

Verwenden Sie den `get-role`-Befehl, um die einer Rolle angefügten Vertrauensrichtlinie anzuzeigen. Verwenden Sie den `get-role-policy`-Befehl, um die Details einer Berechtigungsrichtlinie anzuzeigen.

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listet Sie die Richtlinien auf.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Liste der Namen von Inline-Richtlinien zurückgegeben, die in die IAM-Rolle **`lamda_exec_role`** eingebettet sind. Verwenden Sie den Befehl**`Get-IAMRolePolicy`**, um die Details einer Inline-Richtlinie anzuzeigen.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Ausgabe:

```
oneClick_lambda_exec_role_policy
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS Tools for PowerShell Cmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
```

```
        raise
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS API-Referenz zum SDK for Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien in der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien im AWS SDK für die Rust-API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func listRolePolicies(role: String) async throws -> [String] {
    var policyList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolePoliciesInput(
            marker: marker,
            roleName: role
        )
        let output = try await client.listRolePolicies(input: input)

        guard let policies = output.policyNames else {
            return policyList
        }

        for policy in policies {
            policyList.append(policy)
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return policyList
}
```

- Einzelheiten zur API finden Sie unter ListRoleRichtlinien im AWS SDK für die Swift-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListRoleTags`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListRoleTags`.

CLI

AWS CLI

Um die einer Rolle angehängten Tags aufzulisten

Mit dem folgenden `list-role-tags` Befehl wird die Liste der Tags abgerufen, die der angegebenen Rolle zugeordnet sind.

```
aws iam list-role-tags \
    --role-name production-role
```

Ausgabe:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Weitere Informationen finden Sie unter Tagging IAM-Ressourcen im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListRoleTags in AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das der Rolle zugeordnete Tag abgerufen..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Einzelheiten zur API finden Sie unter Referenz zu ListRoleTags in AWS Tools for PowerShell Cmdlets.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListRoles`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListRoles`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- Einzelheiten zur API finden Sie ListRolesin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So listen Sie die IAM-Rollen für das aktuelle Konto auf

Der folgende `list-roles`-Befehl listet die IAM-Rollen für das aktuelle Konto auf.

```
aws iam list-roles
```

Ausgabe:

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
```

```
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListRolesin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
```

```
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Einzelheiten zur API finden Sie ListRolesin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Rollen auf.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
```

```
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Einzelheiten zur API finden Sie ListRolesin der AWS SDK for JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listRolesArguments["MaxItems"] = $maxItems;
        }
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Einzelheiten zur API finden Sie ListRolesin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Liste aller IAM-Rollen in der abgerufen. AWS-Konto

```
Get-IAMRoleList
```

Beispiel 2: In diesem Beispielcodeausschnitt wird eine Liste von IAM-Rollen im AWS Konto abgerufen und jeweils drei Rollen angezeigt. Anschließend wird darauf gewartet, dass Sie zwischen den einzelnen Gruppen die Eingabetaste drücken. Es übergibt den **`Marker`** Wert des vorherigen Aufrufs, um anzugeben, wo die nächste Gruppe beginnen soll.

```
$nextMarker = $null
Do
{
    $results = Get-IAMRoleList -MaxItem 3 -Marker $nextMarker
    $nextMarker = $AWSHistory.LastServiceResponse.Marker
    $results
    Read-Host
} while ($nextMarker -ne $null)
```

- Einzelheiten zur API finden Sie unter ListRoles AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
```

```
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Einzelheiten zur API finden Sie ListRolesin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count
      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
      roles_counted += 1
    end
    break if roles_counted >= count
  end
```

```
    role_names
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't list roles for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie ListRolesin der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Einzelheiten zur API finden Sie ListRolesin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func listRoles() async throws -> [String] {
    var roleList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolesInput(marker: marker)
        let output = try await client.listRoles(input: input)

        guard let roles = output.roles else {
            return roleList
        }

        for role in roles {
            if let name = role.roleName {
                roleList.append(name)
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return roleList
}
```

- Einzelheiten zur API finden Sie ListRolesin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListSAMLProviders`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListSAMLProviders`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz für AWS SDK for .NET .

CLI

AWS CLI

Um die SAML-Anbieter im Konto aufzulisten AWS

In diesem Beispiel wird die Liste der SAML 2.0-Anbieter abgerufen, die im aktuellen Konto erstellt wurde. AWS

```
aws iam list-saml-providers
```

Ausgabe:

```
{
    "SAMLProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-SAML-Identitätsanbietern im AWS -IAM-Benutzerhandbuch.

- API-Details finden Sie unter ListSAMLProviders in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz für AWS SDK for Go .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die SAML-Anbieter auf.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz für AWS SDK for JavaScript .

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz für AWS SDK for PHP .

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Liste der SAML 2.0-Anbieter abgerufen, die in der aktuellen Version erstellt wurden. AWS-Konto Es gibt den ARN, das Erstellungs- und das Ablaufdatum für jeden SAML-Anbieter zurück.

```
Get-IAMSAMLProviderList
```

Ausgabe:

```
Arn                                                  CreateDate
   ValidUntil
---                                                  ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS     12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Einzelheiten zur API finden Sie unter ListSamlProviders in der Cmdlet-Referenz.AWS Tools for PowerShell

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
```

```
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz zum AWS -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
```

```
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz für AWS SDK for Ruby .

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Weitere API-Informationen finden Sie unter ListSAMLProviders in der API-Referenz zum AWS -SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListServerCertificates`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListServerCertificates`.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::listServerCertificates(
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();
```

```
        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
                      certificate.GetArn() << std::setw(14) <<

 certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::setw(14) <<

 certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Einzelheiten zur API finden Sie unter ListServerZertifikate in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um die Serverzertifikate in Ihrem AWS Konto aufzulisten

Der folgende `list-server-certificates` Befehl listet alle Serverzertifikate auf, die in Ihrem AWS Konto gespeichert sind und zur Verwendung verfügbar sind.

```
aws iam list-server-certificates
```

Ausgabe:

```
{
```

```
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwaltung von Serverzertifikaten in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListServerZertifikate in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Zertifikate auf.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter ListServerZertifikate in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });
```

```
// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter ListServerZertifikate in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Liste der Serverzertifikate abgerufen, die auf den aktuellen AWS-Konto Server hochgeladen wurden.

```
Get-IAMServerCertificateList
```

Ausgabe:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Einzelheiten zur API finden Sie unter ListServerCertificates in AWS Tools for PowerShell Cmdlet Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Serverzertifikate auflisten, aktualisieren und löschen.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
```

```
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter ListServerZertifikate in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListSigningCertificates`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListSigningCertificates`.

CLI

AWS CLI

Um die Signaturzertifikate für einen IAM-Benutzer aufzulisten

Der folgende `list-signing-certificates` Befehl listet die Signaturzertifikate für den genannten IAM-Benutzer auf. `Bob`

```
aws iam list-signing-certificates \
    --user-name Bob
```

Ausgabe:

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

Weitere Informationen finden Sie unter Signaturzertifikate verwalten im Amazon EC2 EC2-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ListSigningZertifikate in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden Details über das Signaturzertifikat abgerufen, das dem genannten **`Bob`** Benutzer zugeordnet ist.

```
Get-IAMSigningCertificate -UserName Bob
```

Ausgabe:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                   YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                   21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                   rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                   nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                   NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                   -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Einzelheiten zur API finden Sie unter ListSigningCertificates in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListUserPolicies`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListUserPolicies`.

CLI

AWS CLI

So listen Sie Richtlinien für einen IAM-Benutzer auf

Der folgende `list-user-policies`-Befehl listet die Richtlinien auf, die dem IAM-Benutzer mit dem Namen Bob zugeordnet sind.

```
aws iam list-user-policies \
    --user-name Bob
```

Ausgabe:

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Einen IAM-Benutzer in Ihrem AWS Konto erstellen.AWS

- Einzelheiten zur API finden Sie unter ListUserRichtlinien in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Einzelheiten zur API finden Sie unter ListUserRichtlinien in der AWS SDK for Go API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Liste der Namen der Inline-Richtlinien abgerufen, die in den IAM-Benutzer namens eingebettet sind. **`David`**

```
Get-IAMUserPolicyList -UserName David
```

Ausgabe:

```
Davids_IAM_Admin_Policy
```

- Einzelheiten zur API finden Sie unter ListUserRichtlinien in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListUserTags`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListUserTags`.

CLI

AWS CLI

Um die einem Benutzer zugewiesenen Tags aufzulisten

Mit dem folgenden `list-user-tags` Befehl wird die Liste der Tags abgerufen, die dem angegebenen IAM-Benutzer zugeordnet sind.

```
aws iam list-user-tags \
    --user-name alice
```

Ausgabe:

```
{
```

```
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Weitere Informationen finden Sie unter Tagging IAM-Ressourcen im IAM-Benutzerhandbuch.AWS

- Einzelheiten zur API finden Sie unter ListUserTags in AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das dem Benutzer zugeordnete Tag abgerufen.

```
Get-IAMUserTagList -UserName joe
```

- Einzelheiten zur API finden Sie unter Referenz zu ListUserTags in AWS Tools for PowerShell Cmdlets.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListUsers`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListUsers`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
########################################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
```

```
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So listen Sie IAM Benutzer auf

Der folgende `list-users`-Befehl listet die IAM-Benutzer im aktuellen Konto auf.

```
aws iam list-users
```

Ausgabe:

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
}
```

Weitere Informationen finden Sie unter Auflisten von IAM-Benutzern im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListUsersin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
```

```
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
 ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
 user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

 permissionsBoundary.permissionsBoundaryTypeAsString());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Benutzer auf.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);
  response.Users?.forEach(({ UserName, CreateDate }) => {
    console.log(`${UserName} created on: ${CreateDate}`);
  });
  return response;
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
```

```
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listAllUsers() {
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
```

```
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- API-Details finden Sie ListUsersin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
$uuid = uniqid();
$service = new IAMService();

    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for PHP API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Sammlung von Benutzern in der aktuellen AWS-Konto Version abgerufen.

```
Get-IAMUserList
```

Ausgabe:

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator

Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : L3EWNONDOM3YUEXAMPLE2
UserName         : bab

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Einzelheiten zur API finden Sie unter ListUsers AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Einzelheiten zur API finden Sie ListUsersin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
  # Lists all users in the AWS account
  #
  # @return [Array<Aws::IAM::Types::User>] An array of user objects
  def list_users
    users = []
    @iam_client.list_users.each_page do |page|
      page.users.each do |user|
        users << user
      end
    end
    users
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing users: #{e.message}")
    []
  end
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
```

```
        .await?;
    Ok(response)
}
```

- Einzelheiten zur API finden Sie ListUsersin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListUsersInput(marker: marker)
            let output = try await client.listUsers(input: input)

            guard let users = output.users else {
                return userList
            }

            for user in users {
                if let id = user.userId, let name = user.userName {
                    userList.append(MyUserRecord(id: id, name: name))
                }
```

```
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return userList
    }
```

- Einzelheiten zur API finden Sie ListUsersin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListVirtualMfaDevices`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListVirtualMfaDevices`.

CLI

AWS CLI

Um virtuelle MFA-Geräte aufzulisten

Der folgende `list-virtual-mfa-devices` Befehl listet die virtuellen MFA-Geräte auf, die für das aktuelle Konto konfiguriert wurden.

```
aws iam list-virtual-mfa-devices
```

Ausgabe:

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

Weitere Informationen finden Sie unter Aktivieren eines Geräts mit virtueller Multi-Faktor-Authentifizierung (MFA) im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie ListVirtualMfaDevicesin AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Sammlung der virtuellen MFA-Geräte abgerufen, die Benutzern im AWS Konto zugewiesen sind. Bei jeder **`User`** Eigenschaft handelt es sich um ein Objekt mit Angaben zum IAM-Benutzer, dem das Gerät zugewiesen ist.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Ausgabe:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Einzelheiten zur API finden Sie unter ListVirtualMfaDevices AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutGroupPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutGroupPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   /// <summary>
   /// Add or update an inline policy document that is embedded in an IAM group.
   /// </summary>
   /// <param name="groupName">The name of the IAM group.</param>
   /// <param name="policyName">The name of the IAM policy.</param>
   /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
   {
       var request = new PutGroupPolicyRequest
       {
           GroupName = groupName,
           PolicyName = policyName,
           PolicyDocument = policyDocument
       };

       var response = await _IAMService.PutGroupPolicyAsync(request);
       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }
```

- Einzelheiten zur API finden Sie unter PutGroupRichtlinie in der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So fügen Sie eine Richtlinie zu einer Gruppe hinzu

Der folgende `put-group-policy`-Befehl fügt eine Richtlinie zur IAM-Gruppe mit dem Namen `Admins` hinzu.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Die Richtlinie ist als JSON-Dokument in der AdminPolicyJSON-Datei definiert. (Der Dateiname und die Erweiterung sind nicht von Bedeutung.)

Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter PutGroupRichtlinie in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Inline-Richtlinie mit dem Namen erstellt **`AppTesterPolicy`** und in die IAM-Gruppe eingebettet. **`AppTesters`** Wenn bereits eine Inline-Richtlinie mit demselben Namen existiert, wird sie überschrieben. Der Inhalt der JSON-Richtlinie wird in der Datei **`apptesterpolicy.json`** gespeichert. Beachten Sie, dass Sie den **`-Raw`** Parameter verwenden müssen, um den Inhalt der JSON-Datei erfolgreich zu verarbeiten.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Einzelheiten zur API finden Sie unter PutGroupRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutRolePermissionsBoundary`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutRolePermissionsBoundary`.

CLI

AWS CLI

Beispiel 1: Um eine auf einer benutzerdefinierten Richtlinie basierende Berechtigungsgrenze auf eine IAM-Rolle anzuwenden

Im folgenden `put-role-permissions-boundary` Beispiel wird die benutzerdefinierte Richtlinie angewendet, die `intern-boundary` als Berechtigungsgrenze für die angegebene IAM-Rolle bezeichnet wird.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Beispiel 2: Um eine auf einer AWS verwalteten Richtlinie basierende Berechtigungsgrenze auf eine IAM-Rolle anzuwenden

Im folgenden `put-role-permissions-boundary` Beispiel AWS wird die verwaltete `PowerUserAccess` Richtlinie als Berechtigungsgrenze für die angegebene IAM-Rolle angewendet.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Ändern einer Rolle im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter PutRolePermissionsBoundary AWS CLIBefehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel zeigt, wie die Berechtigungsgrenze für eine IAM-Rolle festgelegt wird. Sie können AWS verwaltete Richtlinien oder benutzerdefinierte Richtlinien als Berechtigungsgrenze festlegen.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Einzelheiten zur API finden Sie unter PutRolePermissionsBoundary AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutRolePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutRolePolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie unter PutRoleRichtlinie in der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);
```

```
    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie unter PutRoleRichtlinie in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So fügen Sie einer IAM-Rolle eine Berechtigungsrichtlinie hinzu

Der folgende `put-role-policy`-Befehl fügt der Rolle mit dem Namen `Test-Role` eine Berechtigungsrichtlinie hinzu.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Die Richtlinie ist als JSON-Dokument in der AdminPolicyJSON-Datei definiert. (Der Dateiname und die Erweiterung sind nicht von Bedeutung.)

Verwenden Sie den `update-assume-role-policy`-Befehl, um einer Rolle eine Vertrauensrichtlinie anzufügen.

Weitere Informationen finden Sie unter Ändern einer Rolle im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter PutRoleRichtlinie in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::some-test-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
});
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Einzelheiten zur API finden Sie unter PutRoleRichtlinie in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Inline-Richtlinie mit dem Namen erstellt **`FedTesterRolePolicy`** und in die IAM-Rolle eingebettet. **`FedTesterRole`** Wenn bereits eine Inline-Richtlinie mit demselben Namen existiert, wird sie überschrieben. Der Inhalt der JSON-Richtlinie stammt aus der Datei**`FedTesterPolicy.json`**. Beachten Sie, dass Sie den **`-Raw`** Parameter verwenden müssen, um den Inhalt der JSON-Datei erfolgreich zu verarbeiten.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Einzelheiten zur API finden Sie unter PutRoleRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutUserPermissionsBoundary`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutUserPermissionsBoundary`.

CLI

AWS CLI

Beispiel 1: Um eine auf einer benutzerdefinierten Richtlinie basierende Berechtigungsgrenze auf einen IAM-Benutzer anzuwenden

Im folgenden `put-user-permissions-boundary` Beispiel wird eine benutzerdefinierte Richtlinie angewendet, die `intern-boundary` als Berechtigungsgrenze für den angegebenen IAM-Benutzer bezeichnet wird.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Beispiel 2: Um eine auf einer AWS verwalteten Richtlinie basierende Berechtigungsgrenze auf einen IAM-Benutzer anzuwenden

Im folgenden `put-user-permissions-boundary` Beispiel AWS wird die verwaltete Richtlinie angewendet, die `PowerUserAccess` als Berechtigungsgrenze für den angegebenen IAM-Benutzer bezeichnet wird.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Informationen finden Sie im Abschnitt Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter PutUserPermissionsBoundary AWS CLIBefehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Dieses Beispiel zeigt, wie die Berechtigungsgrenze für den Benutzer festgelegt wird. Sie können AWS verwaltete Richtlinien oder benutzerdefinierte Richtlinien als Berechtigungsgrenze festlegen.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Einzelheiten zur API finden Sie unter PutUserPermissionsBoundary AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutUserPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutUserPolicy`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen Sie einen Benutzer und nehmen Sie eine Rolle an

CLI

AWS CLI

So fügen Sie einem IAM-Benutzer eine Richtlinie an

Der folgende `put-user-policy`-Befehl fügt dem IAM-Benutzer mit dem Namen `Bob` eine Richtlinie an.

```
aws iam put-user-policy \
```

```
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Die Richtlinie ist als JSON-Dokument in der AdminPolicyJSON-Datei definiert. (Der Dateiname und die Erweiterung sind nicht von Bedeutung.)

Informationen finden Sie im Abschnitt Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter PutUserRichtlinie in der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
```

```
 roleArn string) error {
policyDoc := PolicyDocument{
 Version:   "2012-10-17",
 Statement: []PolicyStatement{{
  Effect: "Allow",
  Action: actions,
  Resource: aws.String(roleArn),
 }},
}
policyBytes, err := json.Marshal(policyDoc)
if err != nil {
 log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
err)
 return err
}
_, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
&iam.PutUserPolicyInput{
 PolicyDocument: aws.String(string(policyBytes)),
 PolicyName:     aws.String(policyName),
 UserName:       aws.String(userName),
})
if err != nil {
 log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
err)
}
 return err
}
```

- Einzelheiten zur API finden Sie unter PutUserRichtlinie in der AWS SDK for Go API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird eine Inline-Richtlinie mit dem Namen erstellt **`EC2AccessPolicy`** und in den IAM-Benutzer eingebettet. **`Bob`** Wenn bereits eine Inline-Richtlinie mit demselben Namen existiert, wird sie überschrieben. Der Inhalt der JSON-Richtlinie stammt aus der Datei**`EC2AccessPolicy.json`**. Beachten Sie, dass Sie den **`-Raw`** Parameter verwenden müssen, um den Inhalt der JSON-Datei erfolgreich zu verarbeiten.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Einzelheiten zur API finden Sie unter PutUserRichtlinie in der AWS Tools for PowerShell Cmdlet-Referenz.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Creates an inline policy for a specified user.
 # @param username [String] The name of the IAM user.
 # @param policy_name [String] The name of the policy to create.
 # @param policy_document [String] The JSON policy document.
 # @return [Boolean]
 def create_user_policy(username, policy_name, policy_document)
   @iam_client.put_user_policy({
     user_name: username,
     policy_name: policy_name,
     policy_document: policy_document
   })
   @logger.info("Policy #{policy_name} created for user #{username}.")
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   false
 end
```

- Einzelheiten zur API finden Sie unter PutUserRichtlinie in der AWS SDK for Ruby API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   func putUserPolicy(policyDocument: String, policyName: String, user:
IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie unter PutUserRichtlinie im AWS SDK für die Swift-API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`RemoveClientIdFromOpenIdConnectProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`RemoveClientIdFromOpenIdConnectProvider`.

CLI

AWS CLI

Um die angegebene Client-ID aus der Liste der Client-IDs zu entfernen, die für den angegebenen IAM OpenID Connect-Anbieter registriert sind

In diesem Beispiel wird die Client-ID `My-TestApp-3` aus der Liste der Client-IDs entfernt, die dem IAM-OIDC-Anbieter zugeordnet sind, dessen ARN lautet. `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie in der Befehlsreferenz RemoveClientIdFromOpenIdConnectProvider.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Client-ID **`My-TestApp-3`** aus der Liste der Client-IDs entfernt, die dem IAM-OIDC-Anbieter zugeordnet sind, dessen ARN lautet. **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`**

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Einzelheiten zur API finden Sie unter RemoveClientIdFromOpenIdConnectProviderCmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`RemoveRoleFromInstanceProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`RemoveRoleFromInstanceProfile`.

CLI

AWS CLI

Um eine Rolle aus einem Instanzprofil zu entfernen

Mit dem folgenden `remove-role-from-instance-profile` Befehl wird die angegebene Rolle `Test-Role` aus dem genannten Instanzprofil entfernt`ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

Weitere Informationen finden Sie unter Verwenden von Instance-Profilen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter RemoveRoleFromInstanceProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die angegebene Rolle **`MyNewRole`** aus dem genannten EC2-Instanzprofil gelöscht. **`MyNewRole`** Ein Instanzprofil, das in der IAM-Konsole erstellt wird, hat immer denselben Namen wie die Rolle, wie in diesem Beispiel. Wenn Sie sie in der API oder CLI erstellen, können sie unterschiedliche Namen haben.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Einzelheiten zur API finden Sie unter RemoveRoleFromInstanceProfil in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`RemoveUserFromGroup`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`RemoveUserFromGroup`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen einer Benutzergruppe und Hinzufügen eines Benutzers

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
```

```
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie RemoveUserFromGroupin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So entfernen Sie einen Benutzer aus einer IAM-Gruppe

Mit dem folgenden `remove-user-from-group`-Befehl wird der Benutzer mit dem Namen `Bob` aus der IAM-Gruppe mit dem Namen `Admins` entfernt.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Hinzufügen und Entfernen von Benutzern in einer IAM-Benutzergruppe im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie RemoveUserFromGroupin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der IAM-Benutzer **`Bob`** aus der Gruppe **`Testers`** entfernt.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Beispiel 2: In diesem Beispiel werden alle Gruppen gefunden, in denen der IAM-Benutzer Mitglied **Theresa** ist, und entfernt sie dann **Theresa** aus diesen Gruppen.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Beispiel 3: Dieses Beispiel zeigt eine alternative Möglichkeit, den IAM-Benutzer **Bob** aus der **Testers** Gruppe zu entfernen.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Einzelheiten zur API finden Sie unter RemoveUserFromGroup AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **ResyncMfaDevice** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ResyncMfaDevice`.

CLI

AWS CLI

So synchronisieren Sie ein MFA-Gerät

Im folgenden `resync-mfa-device` Beispiel wird das MFA-Gerät synchronisiert, das dem IAM-Benutzer zugeordnet ist `Bob` und dessen ARN `arn:aws:iam::123456789012:mfa/BobsMFADevice` mit einem Authentifizierungsprogramm verknüpft ist, das die beiden Authentifizierungscodes bereitgestellt hat.

```
aws iam resync-mfa-device \
    --user-name Bob \
```

```
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter ResyncMfaDevice in Command Reference.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das MFA-Gerät synchronisiert, das dem IAM-Benutzer zugeordnet ist **Bob** und dessen ARN **arn:aws:iam::123456789012:mfa/bob** mit einem Authentifizierungsprogramm verknüpft ist, das die beiden Authentifizierungscodes bereitgestellt hat.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Beispiel 2: In diesem Beispiel wird das IAM-MFA-Gerät, das dem IAM-Benutzer zugeordnet ist, **Theresa** mit einem physischen Gerät synchronisiert, das die Seriennummer hat **ABCD12345678** und das die beiden Authentifizierungscodes bereitgestellt hat.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Einzelheiten zur API finden Sie unter ResyncMfaDevice in Cmdlet Reference.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **SetDefaultPolicyVersion** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`SetDefaultPolicyVersion`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Verwalten von Richtlinien
- Zurücksetzen einer Richtlinienversion

CLI

AWS CLI

Um die angegebene Version der angegebenen Richtlinie als Standardversion der Richtlinie festzulegen.

In diesem Beispiel wird die v2 Version der Richtlinie festgelegt, deren ARN `arn:aws:iam::123456789012:policy/MyPolicy` die aktive Standardversion ist.

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie SetDefaultPolicyVersionunter AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die **v2** Version der Richtlinie festgelegt, deren ARN die aktive Standardversion ist**arn:aws:iam::123456789012:policy/MyPolicy**.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Einzelheiten zur API finden Sie unter SetDefaultPolicyVersion AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`TagRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`TagRole`.

CLI

AWS CLI

Um einer Rolle ein Tag hinzuzufügen

Der folgende `tag-role` Befehl fügt der angegebenen Rolle ein Tag mit einem Abteilungsnamen hinzu.

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Tagging IAM-Ressourcen im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie TagRolein AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der Rolle im Identity Management Service ein Tag hinzugefügt

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Einzelheiten zur API finden Sie unter TagRole AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`TagUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`TagUser`.

CLI

AWS CLI

Um einem Benutzer ein Tag hinzuzufügen

Mit dem folgenden `tag-user` Befehl wird dem angegebenen Benutzer ein Tag mit der zugehörigen Abteilung hinzugefügt.

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Tagging IAM-Ressourcen im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter Befehlsreferenz TagUser.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird dem Benutzer im Identity Management Service ein Tag hinzugefügt

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Einzelheiten zur API finden Sie unter TagUser AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UntagRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UntagRole`.

CLI

AWS CLI

Um ein Tag aus einer Rolle zu entfernen

Mit dem folgenden `untag-role` Befehl werden alle Tags mit dem Schlüsselnamen „Department“ aus der angegebenen Rolle entfernt.

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Tagging IAM-Ressourcen im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UntagRolein AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Tag aus der Rolle „MyRoleName“ mit dem Tag-Schlüssel „abac“ entfernt. Um mehrere Tags zu entfernen, stellen Sie eine durch Kommas getrennte Tag-Schlüsselliste bereit.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Einzelheiten zur API finden Sie unter UntagRole AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UntagUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UntagUser`.

CLI

AWS CLI

Um ein Tag von einem Benutzer zu entfernen

Mit dem folgenden `untag-user` Befehl werden alle Tags mit dem Schlüsselnamen „Department“ vom angegebenen Benutzer entfernt.

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Tagging IAM-Ressourcen im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter Befehlsreferenz UntagUser.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Tag vom Benutzer mit dem Namen „joe“ und dem Tag-Schlüssel „abac“ und „xyzw“ entfernt. Um mehrere Tags zu entfernen, geben Sie eine durch Kommas getrennte Liste der Tag-Schlüssel an.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Einzelheiten zur API finden Sie unter UntagUser AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateAccessKey`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateAccessKey`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffsschlüsseln

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
```

```
}
```

- Einzelheiten zur API finden Sie unter UpdateAccessSchlüssel in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So aktivieren oder deaktivieren Sie einen Zugriffsschlüssel für einen IAM-Benutzer

Mit dem folgenden `update-access-key`-Befehl wird der angegebene Zugriffsschlüssel (Zugriffsschlüssel-ID und geheimer Zugriffsschlüssel) für den IAM-Benutzer mit dem Namen Bob deaktiviert.

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Die Deaktivierung des Schlüssels bedeutet, dass er nicht für den programmatischen Zugriff auf verwendet werden kann. AWS Der Schlüssel ist jedoch weiterhin verfügbar und kann erneut aktiviert werden.

Weitere Informationen finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateAccessKey in AWS CLI Command Reference.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessId> <status>\s

                Where:
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
 String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
            System.out.printf("Successfully updated the status of access key %s
 to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie unter UpdateAccessSchlüssel in der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Aktualisieren Sie den Zugriffsschlüssel.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter UpdateAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter UpdateAccessSchlüssel in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der Status des Zugriffsschlüssels **`AKIAIOSFODNN7EXAMPLE`** für den IAM-Benutzer mit dem Namen **`Bob`** to **`Inactive`** geändert.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Einzelheiten zur API finden Sie unter UpdateAccessKey in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise
```

- Einzelheiten zur API finden Sie unter UpdateAccessKey in AWS SDK for Python (Boto3) API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateAccountPasswordPolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateAccountPasswordPolicy`.

CLI

AWS CLI

Um die Passwortrichtlinie für das aktuelle Konto festzulegen oder zu ändern

Mit dem folgenden `update-account-password-policy` Befehl wird für die Kennwortrichtlinie eine Mindestlänge von acht Zeichen und eine oder mehrere Zahlen im Kennwort festgelegt.

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Änderungen an der Passwortrichtlinie eines Kontos wirken sich auf alle neuen Passwörter aus, die für IAM-Benutzer im Konto erstellt werden. Änderungen der Passwortrichtlinie wirken sich nicht auf bestehende Passwörter aus.

Weitere Informationen finden Sie unter Festlegen einer Kontopasswortrichtlinie für IAM-Benutzer im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UpdateAccountPasswordPolicyin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Kennwortrichtlinie für das Konto mit den angegebenen Einstellungen aktualisiert. Beachten Sie, dass alle Parameter, die nicht im Befehl enthalten sind, nicht unverändert bleiben. Stattdessen werden sie auf die Standardwerte zurückgesetzt.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Einzelheiten zur API finden Sie unter UpdateAccountPasswordPolicy AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateAssumeRolePolicy`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateAssumeRolePolicy`.

CLI

AWS CLI

Um die Vertrauensrichtlinie für eine IAM-Rolle zu aktualisieren

Der folgende `update-assume-role-policy` Befehl aktualisiert die Vertrauensrichtlinie für die angegebene `Test-Role` Rolle.

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Die Vertrauensrichtlinie ist als JSON-Dokument in der Datei Test-Role-Trust-Policy.json definiert. (Der Dateiname und die Erweiterung sind nicht von Bedeutung.) Die Vertrauensrichtlinie muss einen Prinzipal angeben.

Verwenden Sie den `put-role-policy` Befehl, um die Berechtigungsrichtlinie für eine Rolle zu aktualisieren.

Weitere Informationen finden Sie unter Erstellen von IAM-Rollen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UpdateAssumeRolePolicyunter AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die benannte IAM-Rolle **`ClientRole`** mit einer neuen Vertrauensrichtlinie aktualisiert, deren Inhalt aus der Datei **`ClientRolePolicy.json`** stammt. Beachten Sie, dass Sie den **`-Raw`** Switch-Parameter verwenden müssen, um den Inhalt der JSON-Datei erfolgreich zu verarbeiten.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Einzelheiten zur API finden Sie unter UpdateAssumeRolePolicy AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateGroup`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateGroup`.

CLI

AWS CLI

Um eine IAM-Gruppe umzubenennen

Mit dem folgenden `update-group` Befehl wird der Name der IAM-Gruppe `Test` in geändert. `Test-1`

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Umbenennen einer IAM-Benutzergruppe im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UpdateGroupin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die IAM-Gruppe **`Testers`** in umbenannt. **`AppTesters`**

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Beispiel 2: In diesem Beispiel wird der Pfad der IAM-Gruppe in geändert. **`AppTesters`** **`/Org1/Org2/`** Dadurch wird der ARN für die Gruppe auf geändert**`arn:aws:iam::123456789012:group/Org1/Org2/AppTesters`**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Einzelheiten zur API finden Sie unter UpdateGroup AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateLoginProfile`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateLoginProfile`.

CLI

AWS CLI

Um das Passwort für einen IAM-Benutzer zu aktualisieren

Der folgende `update-login-profile` Befehl erstellt ein neues Passwort für den IAM-Benutzer mit dem Namen. `Bob`

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Verwenden Sie den `update-account-password-policy` Befehl, um eine Kennwortrichtlinie für das Konto festzulegen. Wenn das neue Passwort gegen die Passwortrichtlinie für das Konto verstößt, gibt der Befehl einen `PasswordPolicyViolation` Fehler zurück.

Wenn die Kontopasswortrichtlinie dies zulässt, können IAM-Benutzer ihre eigenen Passwörter mithilfe des `change-password` Befehls ändern.

Bewahren Sie das Passwort an einem sicheren Ort auf. Wenn das Passwort verloren geht, kann es nicht wiederhergestellt werden, und Sie müssen mit dem `create-login-profile` Befehl ein neues erstellen.

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Passwörter für AWS IAM-Benutzer verwalten.

- Einzelheiten zur API finden Sie unter UpdateLoginProfile in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein neues temporäres Passwort für den IAM-Benutzer festgelegt und der Benutzer muss das Passwort ändern**Bob**, wenn er sich das nächste Mal anmeldet.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Einzelheiten zur API finden Sie unter UpdateLoginProfile in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateOpenIdConnectProviderThumbprint`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateOpenIdConnectProviderThumbprint`.

CLI

AWS CLI

Um die bestehende Liste der Fingerabdrücke von Serverzertifikaten durch eine neue Liste zu ersetzen

In diesem Beispiel wird die Zertifikat-Fingerabdruckliste für den OIDC-Anbieter aktualisiert, dessen ARN einen neuen Fingerabdruck verwenden `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com` soll.

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Creating OpenID Connect (OIDC) Identity Providers im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateOpenIdConnectProviderThumbprintBefehlsreferenz.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Zertifikat-Fingerabdruckliste für den OIDC-Anbieter aktualisiert, dessen ARN einen neuen Fingerabdruck verwenden **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** soll. Der OIDC-Anbieter teilt den neuen Wert, wenn sich das dem Anbieter zugeordnete Zertifikat ändert.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Einzelheiten zur API finden Sie unter UpdateOpenIdConnectProviderThumbprint AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateRole`.

CLI

AWS CLI

Um die Beschreibung oder Sitzungsdauer einer IAM-Rolle zu ändern

Mit dem folgenden `update-role` Befehl wird die Beschreibung der IAM-Rolle `production-role` auf 12 Stunden geändert `Main production role` und die maximale Sitzungsdauer wird auf 12 Stunden festgelegt.

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Ändern einer Rolle im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UpdateRolein der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel werden die Rollenbeschreibung und der Wert für die maximale Sitzungsdauer (in Sekunden) aktualisiert, für den eine Rollensitzung angefordert werden kann.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Einzelheiten zur API finden Sie unter UpdateRole AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateRoleDescription`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateRoleDescription`.

CLI

AWS CLI

Um die Beschreibung einer IAM-Rolle zu ändern

Mit dem folgenden `update-role` Befehl wird die Beschreibung der IAM-Rolle `production-role` in geändert. `Main production role`

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

Ausgabe:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

Weitere Informationen finden Sie unter Ändern einer Rolle im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateRoleBeschreibung in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird die Beschreibung einer IAM-Rolle in Ihrem Konto aktualisiert.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Einzelheiten zur API finden Sie unter UpdateRoleBeschreibung in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateSamlProvider`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateSamlProvider`.

CLI

AWS CLI

Um das Metadatendokument für einen vorhandenen SAML-Anbieter zu aktualisieren

In diesem Beispiel wird der SAML-Anbieter in IAM, dessen ARN ist, `arn:aws:iam::123456789012:saml-provider/SAMLADFS` mit einem neuen SAML-Metadatendokument aus der Datei aktualisiert. `SAMLMetaData.xml`

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Ausgabe:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

Weitere Informationen finden Sie unter Erstellen von IAM-SAML-Identitätsanbietern im AWS - IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateSamlAnbieter in AWS CLI der Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der SAML-Anbieter in IAM, dessen ARN lautet, **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`** mit einem neuen SAML-Metadatendokument aus der Datei aktualisiert. **`SAMLMetaData.xml`** Beachten Sie, dass Sie

den **-Raw** Switch-Parameter verwenden müssen, um den Inhalt der JSON-Datei erfolgreich zu verarbeiten.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Einzelheiten zur API finden Sie unter UpdateSamlAnbieter in der AWS Tools for PowerShell Cmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **UpdateServerCertificate** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateServerCertificate`.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                           const Aws::String &newCertificateName,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
```

```
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- Einzelheiten zur API finden Sie unter UpdateServerZertifikat in der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um den Pfad oder Namen eines Serverzertifikats in Ihrem AWS Konto zu ändern

Mit dem folgenden `update-server-certificate`-Befehl wird der Name des Zertifikats von `myServerCertificate` in `myUpdatedServerCertificate` geändert. Außerdem wird der Pfad geändert, `/cloudfront/` sodass der CloudFront Amazon-Service darauf zugreifen kann. Mit diesem Befehl wird keine Ausgabe zurückgegeben. Sie können die Ergebnisse der Aktualisierung anzeigen, indem Sie den `list-server-certificates`-Befehl ausführen.

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwaltung von Serverzertifikaten in IAM im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateServerZertifikat in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Aktualisieren Sie ein Serverzertifikat.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateServerZertifikat in der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie unter UpdateServerZertifikat in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Zertifikat mit dem Namen to umbenannt**`MyServerCertificate`**. **`MyRenamedServerCertificate`**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Beispiel 2: In diesem Beispiel wird das Zertifikat mit dem Namen in den Pfad **`MyServerCertificate`** /Org1/Org2/verschoben. Dadurch wird der ARN für die Ressource auf geändert**`arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate`**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Einzelheiten zur API finden Sie unter UpdateServerCertificate in AWS Tools for PowerShell Cmdlet Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Serverzertifikate auflisten, aktualisieren und löschen.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end
```

```
  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Einzelheiten zur API finden Sie unter UpdateServerZertifikat in der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateSigningCertificate`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateSigningCertificate`.

CLI

AWS CLI

Um ein Signaturzertifikat für einen IAM-Benutzer zu aktivieren oder zu deaktivieren

Der folgende `update-signing-certificate` Befehl deaktiviert das angegebene Signaturzertifikat für den genannten IAM-Benutzer. Bob

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Verwenden Sie den Befehl, um die ID für ein Signaturzertifikat abzurufen. `list-signing-certificates`

Weitere Informationen finden Sie unter Signaturzertifikate verwalten im Amazon EC2 EC2-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter UpdateSigningCertificate in AWS CLI Command Reference.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird das Zertifikat aktualisiert, das dem genannten IAM-Benutzer zugeordnet ist **`Bob`** und dessen Zertifikat-ID es als inaktiv kennzeichnet. **`Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU`**

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Einzelheiten zur API finden Sie unter UpdateSigningCertificate in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateUser`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erstellen von schreibgeschützten und schreib-und lesebereichtigten IAM-Benutzern

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie UpdateUserin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So ändern Sie den Namen eines IAM-Benutzers

Mit dem folgenden `update-user`-Befehl wird der Name des IAM-Benutzers `Bob` in `Robert` geändert.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Umbenennen einer IAM-Benutzergruppe im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UpdateUserin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie UpdateUserin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Aktualisieren Sie den Benutzer.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.

- Einzelheiten zur API finden Sie UpdateUserin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie UpdateUserin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?
) {
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- API-Details finden Sie UpdateUserin der API-Referenz zum AWS SDK für Kotlin.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird der IAM-Benutzer **Bob** in umbenannt. **Robert**

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Beispiel 2: In diesem Beispiel wird der Pfad des IAM-Benutzers **Bob** auf geändert**/Org1/Org2/**, wodurch der ARN für den Benutzer effektiv geändert wird. **arn:aws:iam::123456789012:user/Org1/Org2/bob**

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Einzelheiten zur API finden Sie unter UpdateUser AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Einzelheiten zur API finden Sie UpdateUserin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
  false
end
```

- Einzelheiten zur API finden Sie UpdateUserin der AWS SDK for Ruby API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UploadServerCertificate`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UploadServerCertificate`.

CLI

AWS CLI

Um ein Serverzertifikat auf Ihr AWS Konto hochzuladen

Mit dem folgenden Befehl upload-server-certificate wird ein Serverzertifikat auf Ihr Konto hochgeladen. AWS In diesem Beispiel befindet sich das Zertifikat in der Datei `public_key_cert_file.pem`, der zugehörige private Schlüssel in der Datei `my_private_key.pem` und die von der Zertifizierungsstelle (CA) bereitgestellte Zertifikatskette befindet sich in der `my_certificate_chain_file.pem`-Datei. Wenn der Upload der Datei abgeschlossen ist, ist sie unter dem Namen my verfügbar. ServerCertificate Parameter, die mit `file://` beginnen, weisen den Befehl an, den Inhalt der Datei zu lesen und diesen als Parameterwert anstelle des Dateinamens selbst zu verwenden.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Ausgabe:

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Weitere Informationen finden Sie unter Erstellen, Hochladen und Löschen von Serverzertifikaten im Handbuch zur Verwendung von IAM.

- Einzelheiten zur API finden Sie unter UploadServerZertifikat in der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
```

```
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie unter UploadServerZertifikat in der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein neues Serverzertifikat auf das IAM-Konto hochgeladen. Die Dateien, die den Zertifikatshauptteil, den privaten Schlüssel und (optional) die Zertifikatskette enthalten, müssen alle PEM-codiert sein. Beachten Sie, dass die Parameter den tatsächlichen Inhalt der Dateien und nicht die Dateinamen erfordern. Sie müssen den **-Raw** Switch-Parameter verwenden, um den Dateiinhalt erfolgreich zu verarbeiten.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Ausgabe:

```
Arn                 : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration          : 1/14/2018 9:52:36 AM
Path                : /
ServerCertificateId : ASCAJIEXAMPLE7J7HQZYW
```

```
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Einzelheiten zur API finden Sie unter UploadServerCertificate in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UploadSigningCertificate`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UploadSigningCertificate`.

CLI

AWS CLI

Um ein Signaturzertifikat für einen IAM-Benutzer hochzuladen

Mit dem folgenden `upload-signing-certificate` Befehl wird ein Signaturzertifikat für den IAM-Benutzer mit dem Namen hochgeladen. `Bob`

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

Ausgabe:

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

Das Zertifikat befindet sich in einer Datei namens certificate.pem im PEM-Format.

Weitere Informationen finden Sie unter Erstellen und Hochladen eines Benutzersignaturzertifikats im Handbuch Using IAM.

- Einzelheiten zur API finden Sie unter UploadSigningZertifikat in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: In diesem Beispiel wird ein neues X.509-Signaturzertifikat hochgeladen und es dem IAM-Benutzer mit dem Namen zugeordnet. **`Bob`** Die Datei, die den Zertifikatshauptteil enthält, ist PEM-codiert. Der **`CertificateBody`** Parameter erfordert den tatsächlichen Inhalt der Zertifikatsdatei und nicht den Dateinamen. Sie müssen den **`-Raw`** Switch-Parameter verwenden, um die Datei erfolgreich zu verarbeiten.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Ausgabe:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
```

```
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Einzelheiten zur API finden Sie unter UploadSigningCertificate in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für IAM mit SDKs AWS

Die folgenden Codebeispiele zeigen Ihnen, wie Sie gängige Szenarien in IAM mit SDKs implementieren. AWS Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben durch den Aufruf mehrerer Funktionen innerhalb von IAM erledigen können. Jedes Szenario enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Beispiele

- Erstellen und verwalten Sie einen ausfallsicheren Service mithilfe eines AWS SDK
- Erstellen Sie eine IAM-Gruppe und fügen Sie der Gruppe mithilfe eines SDK einen AWS Benutzer hinzu
- Erstellen Sie einen IAM-Benutzer und übernehmen Sie eine Rolle bei der AWS STS Verwendung eines SDK AWS
- Erstellen Sie IAM-Benutzer mit Schreibzugriff und Lese-/Schreibzugriff mithilfe eines SDK AWS
- Verwaltung von IAM-Zugriffsschlüsseln mithilfe eines SDK AWS
- Verwaltung von IAM-Richtlinien mithilfe eines SDK AWS
- Verwaltung von IAM-Rollen mithilfe eines SDK AWS
- Verwalten Sie Ihr IAM-Konto mit einem SDK AWS

- Rollback einer IAM-Richtlinienversion mithilfe eines SDK AWS
- Arbeiten Sie mit der IAM Policy Builder-API mithilfe eines SDK AWS

Erstellen und verwalten Sie einen ausfallsicheren Service mithilfe eines AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie einen Webservice mit Lastenausgleich erstellen, der Buch-, Film- und Liedempfehlungen zurückgibt. Das Beispiel zeigt, wie der Service auf Fehler reagiert und wie der Service für mehr Ausfallsicherheit umstrukturiert werden kann.

- Verwenden Sie eine Gruppe von Amazon EC2 Auto Scaling, um Amazon Elastic Compute Cloud (Amazon EC2)-Instances basierend auf einer Startvorlage zu erstellen und die Anzahl der Instances in einem bestimmten Bereich zu halten.
- Verarbeiten und verteilen Sie HTTP-Anfragen mit Elastic Load Balancing.
- Überwachen Sie den Zustand von Instances in einer Auto-Scaling-Gruppe und leiten Sie Anfragen nur an fehlerfreie Instances weiter.
- Führen Sie auf jeder EC2-Instance einen Python-Webserver aus, um HTTP-Anfragen zu verarbeiten. Der Webserver reagiert mit Empfehlungen und Zustandsprüfungen.
- Simulieren Sie einen Empfehlungsservice mit einer Amazon DynamoDB-Tabelle.
- Steuern Sie die Antwort des Webservers auf Anfragen und Zustandsprüfungen, indem Sie die AWS Systems Manager Parameter aktualisieren.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
    static async Task Main(string[] args)
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
```

```
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);
```

```
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
```

```
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
```

```
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));
```

```
        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
```

```
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
 your default VPC must\n"
                    + "allows access from this computer. You can either add it
 automatically from this\n"
                    + "example or add it yourself using the AWS Management
 Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound traffic from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
 ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound SSH traffic for debugging from your computer's IP address?"))
                {
```

```
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");
```

```
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
```

```
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
```

```
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );
```

```
        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
 resources? (y/n) "))
        {
            await
 _elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
 _elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
 _autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
 _autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
 _autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Erstellen Sie eine Klasse, die Auto-Scaling- und Amazon-EC2-Aktionen beinhaltet.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
```

```
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
```

```
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
```

```
                                    "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
```

```
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
```

```
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
```

```
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        await CreateKeyPair(_keyPairName);
        await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

        var startServerText = await File.ReadAllTextAsync(startupScriptPath);
        var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

        var amiLatest = await _amazonSsm.GetParameterAsync(
            new GetParameterRequest() { Name = _amiParam });
        var amiId = amiLatest.Parameter.Value;
        var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
            new CreateLaunchTemplateRequest()
            {
                LaunchTemplateName = _launchTemplateName,
```

```
                LaunchTemplateData = new RequestLaunchTemplateData()
                {
                    InstanceType = _instanceType,
                    ImageId = amiId,
                    IamInstanceProfile =
                        new

 LaunchTemplateIamInstanceProfileSpecificationRequest()
                        {
                            Name = _instanceProfileName
                        },
                    KeyName = _keyPairName,
                    UserData = System.Convert.ToBase64String(plainTextBytes)
                }
            });
        return launchTemplateResponse.LaunchTemplate;

    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
 Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
```

```
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
            new DescribeVpcsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("is-default", new List<string>() { "true" })
                }
            });
        return vpcResponse.Vpcs[0];
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
```

```
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        var subnets = new List<Subnet>();
        var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
            new DescribeSubnetsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonClientException)
```

```
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
```

```
                    });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
```

```
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);
```

```
            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
```

```
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
```

```
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
```

```
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
```

```
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
```

```
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }
}
```

Erstellen Sie eine Klasse, die Elastic-Load-Balancing-Aktionen beinhaltet.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
```

```
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }
```

```
        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
```

```
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
 how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
```

```
   /// Create an Elastic Load Balancing load balancer that uses the specified
subnets
   /// and forwards requests to the specified target group.
   /// </summary>
   /// <param name="name">The name for the new load balancer.</param>
   /// <param name="subnetIds">Subnets for the load balancer.</param>
   /// <param name="targetGroup">Target group for forwarded requests.</param>
   /// <returns>The new LoadBalancer object.</returns>
   public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
List<string> subnetIds, TargetGroup targetGroup)
   {
       var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
           new CreateLoadBalancerRequest()
           {
               Name = name,
               Subnets = subnetIds
           });
       var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

       // Wait for load balancer to be available.
       var loadBalancerReady = false;
       while (!loadBalancerReady)
       {
           try
           {
               var describeResponse =
                   await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                       new DescribeLoadBalancersRequest()
                       {
                           Names = new List<string>() { name }
                       });

               var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

               loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
           }
           catch (LoadBalancerNotFoundException)
           {
               loadBalancerReady = false;
           }
```

```
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
```

```
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }
```

```
    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
targetArn });
                Console.WriteLine($"Deleted load balancing target group
{groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Erstellen Sie eine Klasse, die DynamoDB zum Simulieren eines Empfehlungsservices verwendet.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
 configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
```

```
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
```

```
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
 _amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }
```

```
    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Erstellen Sie eine Klasse, die Systems-Manager-Aktionen umschließt.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";
```

```
    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - AttachLoadBalancerTargetGruppen
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchVorlage
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGruppe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil
  - DeleteLaunchVorlage
  - DeleteLoadBalancer
  - DeleteTargetGruppe
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZonen
  - DescribeIamInstanceProfileVerbände
  - DescribeInstances
  - DescribeLoadBalancer
  - DescribeSubnets
  - DescribeTargetGruppen
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileVerband
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;
```

```
   public static final String DASHES = new String(new char[80]).replace("\0",
"-");

   public static void main(String[] args) throws IOException,
InterruptedException {
       Scanner in = new Scanner(System.in);
       Database database = new Database();
       AutoScaler autoScaler = new AutoScaler();
       LoadBalancer loadBalancer = new LoadBalancer();

       System.out.println(DASHES);
       System.out.println("Welcome to the demonstration of How to Build and
Manage a Resilient Service!");
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("A - SETUP THE RESOURCES");
       System.out.println("Press Enter when you're ready to start deploying
resources.");
       in.nextLine();
       deploy(loadBalancer);
       System.out.println(DASHES);
       System.out.println(DASHES);
       System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
       System.out.println("Press Enter when you're ready.");
       in.nextLine();
       demo(loadBalancer);
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("C - DELETE THE RESOURCES");
       System.out.println("""
               This concludes the demo of how to build and manage a resilient
service.
               To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
               that were created for this demo.
               """);

       System.out.println("\n Do you want to delete the resources (y/n)? ");
       String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

       if (userInput.equals("y")) {
```

```
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
```

```
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
```

```
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
```

```
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
```

```
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
       demoChoices(loadBalancer);

       System.out.println(
               """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
       paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

       System.out.println(
               """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
       demoChoices(loadBalancer);

       System.out.println(
               """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
       paramHelper.put(paramHelper.failureResponse, "static");

       System.out.println("""
               Now, sending a GET request to the load balancer endpoint returns
a static response.
```

```
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);
```

```
        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
```

```
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
```

```
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
```

```
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
 target.target().id(),
                                    target.target().port(),
 target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
 health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Erstellen Sie eine Klasse, die Auto-Scaling- und Amazon-EC2-Aktionen beinhaltet.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;
```

```
    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
```

```
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
```

```
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
```

```
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
```

```
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
```

```
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
     * this
     * computer. This can be done by allowing ingress from this computer's IP
     * address. In some situations, such as connecting from a corporate network,
you
     * must instead specify a prefix list ID. You can also temporarily open the
port
     * to
     * any IP address while running this example. If you do, be sure to remove
     * public
     * access when you're done.
     *
     */
    public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
        boolean portIsOpen = false;
        GroupInfo groupInfo = new GroupInfo();
        try {
            Filter filter = Filter.builder()
                    .name("group-name")
                    .values("default")
                    .build();

            Filter filter1 = Filter.builder()
                    .name("vpc-id")
                    .values(VPC)
                    .build();
```

```
            DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                    .filters(filter, filter1)
                    .build();

            DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                    .describeSecurityGroups(securityGroupsRequest);
            String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
            groupInfo.setGroupName(securityGroup);

            for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
```

```
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
     */
    public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
        try {
            AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                    .autoScalingGroupName(asGroupName)
                    .targetGroupARNs(targetGroupARN)
                    .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
            System.out.println("Attached load balancer to " + asGroupName);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Creates an EC2 Auto Scaling group with the specified size.
    public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

        // Get availability zones.
```

```
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());

       String availabilityZones = String.join(",", availabilityZoneNames);
       LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
               .launchTemplateName(templateName)
               .version("$Default")
               .build();

       String[] zones = availabilityZones.split(",");
       CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
               .launchTemplate(specification)
               .availabilityZones(zones)
               .maxSize(groupSize)
               .minSize(groupSize)
               .autoScalingGroupName(autoScalingGroupName)
               .build();

       try {
           getAutoScalingClient().createAutoScalingGroup(groupRequest);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
       return zones;
   }
```

```
   public String getDefaultVPC() {
       // Define the filter.
       Filter defaultFilter = Filter.builder()
               .name("is-default")
               .values("true")
               .build();

       software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
               .builder()
               .filters(defaultFilter)
               .build();

       DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
       return response.vpcs().get(0).vpcId();
   }

   // Gets the default subnets in a VPC for a specified list of Availability
Zones.
   public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
       List<Subnet> subnets = null;
       Filter vpcFilter = Filter.builder()
               .name("vpc-id")
               .values(vpcId)
               .build();

       Filter azFilter = Filter.builder()
               .name("availability-zone")
               .values(availabilityZones)
               .build();

       Filter defaultForAZ = Filter.builder()
               .name("default-for-az")
               .values("true")
               .build();

       DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
               .filters(vpcFilter, azFilter, defaultForAZ)
               .build();

       DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
       subnets = response.subnets();
       return subnets;
```

```
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
```

```
        ListPoliciesRequest listPoliciesRequest =
 ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
 getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
 the policy.

 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }
```

```
        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Erstellen Sie eine Klasse, die Elastic-Load-Balancing-Aktionen beinhaltet.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
```

```
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();

        DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
```

```
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);

        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
```

```
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
```

```
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
```

```
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
 become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
 loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .defaultActions(action)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Erstellen Sie eine Klasse, die DynamoDB zum Simulieren eines Empfehlungsservices verwendet.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
 DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
 e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
```

```
    * hash key named 'MediaType' that defines the type of media recommended,
such
    * as
    * Book or Movie, and a range key named 'ItemId' that, combined with the
    * MediaType,
    * forms a unique identifier for the recommended item.
    */
   public void createTable(String tableName, String fileName) throws IOException
{
       // First check to see if the table exists.
       boolean doesExist = doesTableExist(tableName);
       if (!doesExist) {
           DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
           CreateTableRequest createTableRequest = CreateTableRequest.builder()
                   .tableName(tableName)
                   .attributeDefinitions(
                           AttributeDefinition.builder()
                                   .attributeName("MediaType")
                                   .attributeType(ScalarAttributeType.S)
                                   .build(),
                           AttributeDefinition.builder()
                                   .attributeName("ItemId")
                                   .attributeType(ScalarAttributeType.N)
                                   .build())
                   .keySchema(
                           KeySchemaElement.builder()
                                   .attributeName("MediaType")
                                   .keyType(KeyType.HASH)
                                   .build(),
                           KeySchemaElement.builder()
                                   .attributeName("ItemId")
                                   .keyType(KeyType.RANGE)
                                   .build())
                   .provisionedThroughput(
                           ProvisionedThroughput.builder()
                                   .readCapacityUnits(5L)
                                   .writeCapacityUnits(5L)
                                   .build())
                   .build();

           getDynamoDbClient().createTable(createTableRequest);
           System.out.println("Creating table " + tableName + "...");

           // Wait until the Amazon DynamoDB table is created.
```

```
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
```

```
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Erstellen Sie eine Klasse, die Systems-Manager-Aktionen umschließt.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - AttachLoadBalancerTargetGruppen
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchVorlage
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGruppe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil
  - DeleteLaunchVorlage
  - DeleteLoadBalancer
  - DeleteTargetGruppe
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZonen
  - DescribeIamInstanceProfileVerbände
  - DescribeInstances
  - DescribeLoadBalancer
  - DescribeSubnets
  - DescribeTargetGruppen
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileVerband
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0


import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
```

```
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

Erstellen Sie Schritte, um alle Ressourcen bereitzustellen.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
```

```
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
```

```
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
```

```
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
```

```
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
```

```
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
```

```
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
```

```
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
      // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
```

```
          Version: "$Default",
        },
        MinSize: 3,
        MaxSize: 3,
      }),
    ),
  );
}),
new ScenarioOutput(
  "createdAutoScalingGroup",
  /**
   * @param {{ availabilityZoneNames: string[] }} state
   */
  (state) =>
    MESSAGES.createdAutoScalingGroup
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
      .replace(
        "${AVAILABILITY_ZONE_NAMES}",
        state.availabilityZoneNames.join(", "),
      ),
),
new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
  type: "confirm",
}),
new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
new ScenarioAction("getVpc", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
  const client = new EC2Client({});
  const { Vpcs } = await client.send(
    new DescribeVpcsCommand({
      Filters: [{ Name: "is-default", Values: ["true"] }],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
  state.defaultVpc = Vpcs[0].VpcId;
}),
new ScenarioOutput("gotVpc", (state) =>
  MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
),
new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
new ScenarioAction("getSubnets", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
  const client = new EC2Client({});
```

```
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
```

```
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
```

```
  new ScenarioAction("createListener", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
```

```
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
```

```
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      } else {
        return MESSAGES.noIpRules;
      }
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        return MESSAGES.noIpRules;
      }
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
```

```
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    } else {
      return false;
    }
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
];
```

Erstellen Sie Schritte, um die Demo auszuführen.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { readFileSync } from "node:fs";
import { join } from "node:path";
```

```
import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";
```

```
const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  state.targetHealthDescriptions = TargetHealthDescriptions;
```

```
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
```

```
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
```

```
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
```

```
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      // snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
```

```
      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
```

```
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
```

```
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
```

```
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
```

```
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Erstellen Sie Schritte, um alle Ressourcen zu vernichten.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
```

```
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
```

```
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
    }
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    } else {
      return MESSAGES.deletedKeyPair.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
```

```
    }
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.detachedPolicyFromRole
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
```

```
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    } else {
      return MESSAGES.deletedPolicy.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.removedRoleFromInstanceProfile
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
```

```
    }
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
```

```
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
```

```
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
```

```
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    } else {
      return MESSAGES.deletedLoadBalancer.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    } else {
      return MESSAGES.deletedLoadBalancerTargetGroup.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
```

```
      );
    }
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    } else {
      return MESSAGES.detachedSsmOnlyRoleFromProfile
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
```

```
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    } else {
      return MESSAGES.detachedSsmOnlyCustomRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    } else {
      return MESSAGES.detachedSsmOnlyAWSRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
```

```
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
  }),
```

```
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
```

```
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    } else {
      console.log(err.name);
      throw err;
    }
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
```

```
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for JavaScript -API-Referenz.
  - AttachLoadBalancerTargetGruppen
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchVorlage
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGruppe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil
  - DeleteLaunchVorlage
  - DeleteLoadBalancer
  - DeleteTargetGruppe
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZonen
  - DescribeIamInstanceProfileVerbände
  - DescribeInstances
  - DescribeLoadBalancer
  - DescribeSubnets
  - DescribeTargetGruppen

- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileVerband
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    ):
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

    def deploy(self):
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        print(
```

```
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) to create
several AWS resources\n"
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n"
            "against various kinds of failures.\n\n"
            "Some of the resources create by this demo are:\n"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to start deploying resources.")

        print(
            f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
        print("-" * 88)

        print(
            f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
            f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
            f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
            f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
```

```
            f"run a web server, such as Apache, with least-privileged
credentials.\n"
        )
        print(
            f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
            f"that control the flow of the demo.\n"
        )
        self.autoscaler.create_template(startup_script, instance_policy)
        print("-" * 88)

        print(
            f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            f"Availability Zone."
        )
        zones = self.autoscaler.create_group(3)
        print("-" * 88)
        print(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            "HTTP requests. You can see these instances in the console or
continue with the demo."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue.")

        print(f"Creating variables that control the flow of the demo.\n")
        self.param_helper.reset()

        print(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            "defines how the load balancer connects to instances. The load
balancer provides a\n"
            "single endpoint where clients connect and dispatches requests to
instances in the group.\n"
        )
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.protocol, self.port, vpc["VpcId"]
```

```
        )
        self.loadbalancer.create_load_balancer(
            [subnet["SubnetId"] for subnet in subnets], target_group
        )
        self.autoscaler.attach_load_balancer_target_group(target_group)
        print(f"Verifying access to the load balancer endpoint...")
        lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if not lb_success:
            print(
                "Couldn't connect to the load balancer, verifying that the port
 is open..."
            )
            current_ip_address = requests.get(
                "http://checkip.amazonaws.com"
            ).text.strip()
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                print(
                    "For this example to work, the default security group for
 your default VPC must\n"
                    "allows access from this computer. You can either add it
 automatically from this\n"
                    "example or add it yourself using the AWS Management Console.
\n"
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
```

```
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
 from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        else:
            print(
                "Couldn't get a successful response from the load balancer
 endpoint. Troubleshoot by\n"
                "manually verifying that your VPC and security group are
 configured correctly and that\n"
                "you can successfully make a GET request to the load balancer
 endpoint:\n"
            )
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue with the demo.")

    def demo_choices(self):
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            print("-" * 88)
            print(
                "\nSee the current state of the service by selecting one of the
 following choices:\n"
            )
            choice = q.choose("\nWhich action would you like to take? ", actions)
            print("-" * 88)
            if choice == 0:
                print("Request:\n")
                print(f"GET http://{self.loadbalancer.endpoint()}")
                response = requests.get(f"http://{self.loadbalancer.endpoint()}")
```

```
                print("\nResponse:\n")
                print(f"{response.status_code}")
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                print("\nChecking the health of load balancer targets:\n")
                health = self.loadbalancer.check_target_health()
                for target in health:
                    state = target["TargetHealth"]["State"]
                    print(
                        f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                    )
                    if state != "healthy":
                        print(
                            f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                        )
                print(
                    f"\nNote that it can take a minute or two for the health
check to update\n"
                    f"after changes are made.\n"
                )
            elif choice == 2:
                print("\nOkay, let's move on.")
                print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\nThis part of the demonstration shows how to toggle different parts
of the system\n"
            "to create situations where the web service fails, and shows how
using a resilient\n"
            "architecture can keep the web service running in spite of these
failures."
        )
        print("-" * 88)

        print(
```

```
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nNow, sending a GET request to the load balancer endpoint returns a
failure code. But, the service reports as\n"
            "healthy to the load balancer because shallow health checks don't
check for failure of the recommendation service."
        )
        self.demo_choices()

        print(
            f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
            f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
        )
        self.param_helper.put(self.param_helper.failure_response, "static")
        print(
            f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
            f"The service still reports as healthy because health checks are
still shallow.\n"
        )
        self.demo_choices()

        print("Let's reinstate the recommendation service.\n")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
        print(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n"
            "access the DynamoDB recommendation table.\n"
```

```
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        print(
            f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
            f"bad credentials...\n"
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        print(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n"
            "depending on which instance is selected by the load balancer.\n"
        )
        self.demo_choices()

        print(
            "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
            "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
            "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
            "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
            "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
        )
        print(
            "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
            "and take that instance out of rotation.\n"
```

```
        )
        self.param_helper.put(self.param_helper.health_check, "deep")
        print(
            f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
            f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
            f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
            "the load balancer takes unhealthy instances out of its rotation.\n"
        )
        self.demo_choices()

        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
```

```
            "unhealthy instances, allowing them to fail open and return a static
 response rather than fail\n"
            "closed and report failure to the customer."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
 service.\n"
            "To keep things tidy and to avoid unwanted charges on your account,
 we can clean up all AWS resources\n"
            "that were created for this demo."
        )
        if q.ask("Do you want to clean up all demo resources? (y/n) ",
 q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            self.recommendation.destroy()
        else:
            print(
                "Okay, we'll leave the resources intact.\n"
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            )


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
```

```
    )
    parser.add_argument(
        "--resource_path",
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
 Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
    autoscaler = AutoScaler.from_client(prefix)
    loadbalancer = LoadBalancer.from_client(prefix)
    param_helper = ParameterHelper.from_client(recommendation.table_name)
    runner = Runner(
        args.resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    print("-" * 88)
    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
```

```
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Erstellen Sie eine Klasse, die Auto-Scaling- und Amazon-EC2-Aktionen beinhaltet.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
```

```
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,
            "t3.micro",
            "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
            as_client,
            ec2_client,
            ssm_client,
            iam_client,
        )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
```

```
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")
```

```
        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
```

```
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn


    def get_instance_profile(self, instance_id):
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
```

```
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

```
    def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )


    def create_key_pair(self, key_pair_name):
        """
```

```
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        :return: The newly created key pair.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")


    def delete_key_pair(self):
        """
        Deletes a key pair.

        :param key_pair_name: The name of the key pair to delete.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
        except FileNotFoundError:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        except PermissionError:
            log.info(
                "Inadequate permissions to delete key pair %s.",
self.key_pair_name
            )
        except Exception as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
```

```
    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
```

```
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
```

```
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
            return zones


    def create_group(self, group_size):
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        zones = []
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
```

```
                MaxSize=group_size,
            )
            log.info(
                "Created EC2 Auto Scaling group %s with availability zones %s.",
                self.launch_template_name,
                zones,
            )
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExists":
                log.info(
                    "EC2 Auto Scaling group %s already exists, nothing to do.",
                    self.group_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
                )
        return zones


    def get_instances(self):
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: Data about the instances.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
            )
        else:
            return instance_ids
```

```
    def terminate_instance(self, instance_id):
        """
        Terminates and instances in an EC2 Auto Scaling group. After an instance
is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
            )
            log.info("Terminated instance %s.", instance_id)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

    def attach_load_balancer_target_group(self, lb_target_group):
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forward requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
                f"to auto scaling group {self.group_name}"
            )
```

```
   def _try_terminate_instance(self, inst_id):
       stopping = False
       log.info(f"Stopping {inst_id}.")
       while not stopping:
           try:
               self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                   InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
               )
               stopping = True
           except ClientError as err:
               if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                   log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                   time.sleep(10)
               else:
                   raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

   def _try_delete_group(self):
       """
       Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
       the function waits and retries until the group is successfully deleted.
       """
       stopped = False
       while not stopped:
           try:
               self.autoscaling_client.delete_auto_scaling_group(
                   AutoScalingGroupName=self.group_name
               )
               stopped = True
               log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
           except ClientError as err:
               if (
                   err.response["Error"]["Code"] == "ResourceInUse"
                   or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
               ):
                   log.info(
                       "Some instances are still running. Waiting for them to
stop..."
                   )
```

```
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
 group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
```

```
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]


    def verify_inbound_port(self, vpc, port, ip_address):
        """
        Verify the default security group of the specified VPC allows ingress
from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
you
        must instead specify a prefix list ID. You can also temporarily open the
port to
        any IP address while running this example. If you do, be sure to remove
public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specific VPC, and a value that
indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info("Found default security group %s.", sec_group["GroupId"])
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info("Found inbound rule: %s", ip_perm)
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
```

```
                    if not port_is_open:
                        log.info(
                            "The inbound rule does not appear to be open to
either this computer's IP\n"
                            "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                            ip_address,
                        )
                    else:
                        break
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
            )
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id, port, ip_address):
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
```

```
            raise AutoScalerError(
                f"Couldn't authorize ingress to {sec_group_id} on port {port}
 from {ip_address}: {err}"
            )


    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
 Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

Erstellen Sie eine Klasse, die Elastic-Load-Balancing-Aktionen beinhaltet.

```
class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
 the load balancer.
```

```
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)

    def endpoint(self):
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        if self._endpoint is None:
            try:
                response = self.elb_client.describe_load_balancers(
                    Names=[self.load_balancer_name]
                )
                self._endpoint = response["LoadBalancers"][0]["DNSName"]
            except ClientError as err:
                raise LoadBalancerError(
                    f"Couldn't get the endpoint for load balancer
{self.load_balancer_name}: {err}"
                )
        return self._endpoint


    def create_target_group(self, protocol, port, vpc_id):
        """
        Creates an Elastic Load Balancing target group. The target group
specifies how
```

```
        the load balancer forward requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=self.target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info("Created load balancing target group %s.",
self.target_group_name)
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
            )
        else:
            return target_group


    def delete_target_group(self):
        """
        Deletes the target group.
        """
        done = False
```

```
        while not done:
            try:
                response = self.elb_client.describe_target_groups(
                    Names=[self.target_group_name]
                )
                tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
                self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
                log.info(
                    "Deleted load balancing target group %s.",
self.target_group_name
                )
                done = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "TargetGroupNotFound":
                    log.info(
                        "Load balancer target group %s not found, nothing to
do.",
                        self.target_group_name,
                    )
                    done = True
                elif err.response["Error"]["Code"] == "ResourceInUse":
                    log.info(
                        "Target group not yet released from load balancer,
waiting..."
                    )
                    time.sleep(10)
                else:
                    raise LoadBalancerError(
                        f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
                    )


    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param subnet_ids: A list of subnets to associate with the load balancer.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created load balancer.
```

```
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
            log.info("Load balancer is available!")
            self.elb_client.create_listener(
                LoadBalancerArn=load_balancer["LoadBalancerArn"],
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                "Created listener to forward traffic from load balancer %s to
target group %s.",
                self.load_balancer_name,
                target_group["TargetGroupName"],
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Failed to create load balancer {self.load_balancer_name}"
                f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
            )
        else:
            self._endpoint = load_balancer["DNSName"]
            return load_balancer


    def delete_load_balancer(self):
        """
        Deletes a load balancer.
        """
        try:
```

```
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[self.load_balancer_name])
        except ClientError as err:
            if err.response["Error"]["Code"] == "LoadBalancerNotFound":
                log.info(
                    "Load balancer %s does not exist, nothing to do.",
                    self.load_balancer_name,
                )
            else:
                raise LoadBalancerError(
                    f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
                )


    def verify_load_balancer_endpoint(self):
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.
        """
        success = False
        retries = 3
        while not success and retries > 0:
            try:
                lb_response = requests.get(f"http://{self.endpoint()}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    success = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
```

```
                )
                retries -= 1
                time.sleep(10)
        return success

    def check_target_health(self):
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[self.target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't check health of {self.target_group_name} targets:
 {err}"
            )
        else:
            return health_response["TargetHealthDescriptions"]
```

Erstellen Sie eine Klasse, die DynamoDB zum Simulieren eines Empfehlungsservices verwendet.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name, dynamodb_client):
        """
        :param table_name: The name of the DynamoDB recommendations table.
```

```
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        Creates a DynamoDB table to use a recommendation service. The table has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
```

```
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file):
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self):
        """
        Deletes the recommendations table.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
```

```
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Erstellen Sie eine Klasse, die Systems-Manager-Aktionen umschließt.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        Resets the Systems Manager parameters to starting values for the demo.
```

```
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)
        except ClientError as err:
            raise ParameterHelperError(
                f"Couldn't set parameter {name} to {value}: {err}"
            )
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - AttachLoadBalancerTargetGruppen
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchVorlage
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGruppe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil

- DeleteLaunchVorlage
- DeleteLoadBalancer
- DeleteTargetGruppe
- DescribeAutoScalingGroups
- DescribeAvailabilityZonen
- DescribeIamInstanceProfileVerbände
- DescribeInstances
- DescribeLoadBalancer
- DescribeSubnets
- DescribeTargetGruppen
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileVerband
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen Sie eine IAM-Gruppe und fügen Sie der Gruppe mithilfe eines SDK einen AWS Benutzer hinzu

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen Sie eine Gruppe und gewähren Sie ihr vollständige Amazon-S3-Zugriffsberechtigungen.
- Erstellen Sie einen neuen Benutzer ohne Berechtigungen für den Zugriff auf Amazon S3.
- Fügen Sie den Benutzer der Gruppe hinzu und zeigen Sie, dass er jetzt über Berechtigungen für Amazon S3 verfügt. Bereinigen Sie dann die Ressourcen.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
```

```
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
```

```
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
```

```
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
```

```
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
 { UserName = userName });
        return response.User;
```

```
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
```

```
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
```

```
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
```

```
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}


using Microsoft.Extensions.Configuration;

namespace IAMGroups;

public class IAMGroups
{
```

```
    private static ILogger logger = null!;

    // Represents JSON code for AWS full access policy for Amazon Simple
    // Storage Service (Amazon S3).
    private const string S3FullAccessPolicyDocument = "{" +
        " \"Statement\" : [{" +
            " \"Action\" : [\"s3:*\"]," +
            " \"Effect\" : \"Allow\"," +
            " \"Resource\" : \"*\"" +
        "}]" +
    "}";


    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMGroups>();

        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var groupUserName = configuration["GroupUserName"];
        var groupName = configuration["GroupName"];
        var groupPolicyName = configuration["GroupPolicyName"];
        var groupBucketName = configuration["GroupBucketName"];
```

```
        var wrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayGroupsOverview();
        uiWrapper.PressEnter();

        // Create an IAM group.
        uiWrapper.DisplayTitle("Create IAM group");
        Console.WriteLine("Let's begin by creating a new IAM group.");
        var group = await wrapper.CreateGroupAsync(groupName);

        // Add an inline IAM policy to the group.
        uiWrapper.DisplayTitle("Add policy to group");
        Console.WriteLine("Add an inline policy to the group that allows members
to have full access to");
        Console.WriteLine("Amazon Simple Storage Service (Amazon S3) buckets.");

        await wrapper.PutGroupPolicyAsync(group.GroupName, groupPolicyName,
S3FullAccessPolicyDocument);

        uiWrapper.PressEnter();

        // Now create a new user.
        uiWrapper.DisplayTitle("Create an IAM user");
        Console.WriteLine("Now let's create a new IAM user.");
        var groupUser = await wrapper.CreateUserAsync(groupUserName);

        // Add the new user to the group.
        uiWrapper.DisplayTitle("Add the user to the group");
        Console.WriteLine("Adding the user to the group, which will give the user
the same permissions as the group.");
        await wrapper.AddUserToGroupAsync(groupUser.UserName, group.GroupName);

        Console.WriteLine($"User, {groupUser.UserName}, has been added to the
group, {group.GroupName}.");
        uiWrapper.PressEnter();

        Console.WriteLine("Now that we have created a user, and added the user to
the group, let's create an IAM access key.");

        // Create access and secret keys for the user.
        var accessKey = await wrapper.CreateAccessKeyAsync(groupUserName);
        Console.WriteLine("Key created.");
```

```
        uiWrapper.WaitABit(15, "Waiting for the access key to be ready for
use.");

        uiWrapper.DisplayTitle("List buckets");
        Console.WriteLine("To prove that the user has access to Amazon S3, list
the S3 buckets for the account.");

        var s3Client = new AmazonS3Client(accessKey.AccessKeyId,
accessKey.SecretAccessKey);
        var stsClient = new
AmazonSecurityTokenServiceClient(accessKey.AccessKeyId,
accessKey.SecretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client, stsClient);

        var buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        // Show that the user also has write access to Amazon S3 by creating
        // a new bucket.
        uiWrapper.DisplayTitle("Create a bucket");
        Console.WriteLine("Since group members have full access to Amazon S3,
let's create a bucket.");
        var success = await s3Wrapper.PutBucketAsync(groupBucketName);

        if (success)
        {
            Console.WriteLine($"Successfully created the bucket:
{groupBucketName}.");
        }

        uiWrapper.PressEnter();

        Console.WriteLine("Let's list the user's S3 buckets again to show the new
bucket.");
```

```
        buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
 {bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("First delete the bucket we created.");
        await s3Wrapper.DeleteBucketAsync(groupBucketName);

        Console.WriteLine($"Now remove the user, {groupUserName}, from the group,
 {groupName}.");
        await wrapper.RemoveUserFromGroupAsync(groupUserName, groupName);

        Console.WriteLine("Delete the user's access key.");
        await wrapper.DeleteAccessKeyAsync(accessKey.AccessKeyId, groupUserName);

        // Now we can safely delete the user.
        Console.WriteLine("Now we can delete the user.");
        await wrapper.DeleteUserAsync(groupUserName);

        uiWrapper.PressEnter();

        Console.WriteLine("Now we will delete the IAM policy attached to the
 group.");
        await wrapper.DeleteGroupPolicyAsync(groupName, groupPolicyName);

        Console.WriteLine("Now we delete the IAM group.");
        await wrapper.DeleteGroupAsync(groupName);

        uiWrapper.PressEnter();

        Console.WriteLine("The IAM groups demo has completed.");

        uiWrapper.PressEnter();
    }
}
```

```
namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };
```

```
        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
```

```
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
```

```
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
```

```
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.

- AddUserToGroup
- AttachRolePolitik
- CreateAccessSchlüssel
- CreateGroup
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessSchlüssel
- DeleteGroup
- DeleteGroupPolitik
- DeleteUser
- PutGroupPolitik
- RemoveUserFromGroup

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen Sie einen IAM-Benutzer und übernehmen Sie eine Rolle bei der AWS STS Verwendung eines SDK AWS

Die folgenden Codebeispiele veranschaulichen, wie Sie einen Benutzer erstellen und eine Rolle annehmen lassen.

> **Warning**
>
> Um Sicherheitsrisiken zu vermeiden, sollten Sie IAM-Benutzer nicht zur Authentifizierung verwenden, wenn Sie eigens entwickelte Software entwickeln oder mit echten Daten arbeiten. Verwenden Sie stattdessen den Verbund mit einem Identitätsanbieter wie AWS IAM Identity Center.

- Erstellen Sie einen Benutzer ohne Berechtigungen.
- Erstellen einer Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets für das Konto erteilt.

- Hinzufügen einer Richtlinie, damit der Benutzer die Rolle übernehmen kann.
- Übernehmen Sie die Rolle und listen Sie S3-Buckets mit temporären Anmeldeinformationen auf, und bereinigen Sie dann die Ressourcen.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }
```

```
    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
```

```
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
```

```
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
```

```
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
```

```
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
```

```
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
   /// <summary>
   /// Delete an IAM user policy.
   /// </summary>
   /// <param name="policyName">The name of the IAM policy to delete.</param>
   /// <param name="userName">The username of the IAM user.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
   {
       var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Gets the IAM password policy for an AWS account.
   /// </summary>
   /// <returns>The PasswordPolicy for the AWS account.</returns>
   public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
```

```
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
```

```
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }
```

```
        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
```

```
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
```

```
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
```

```
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}
```

```
using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();
```

```
        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);
```

```
        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
```

```
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
```

```
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
```

```
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
```

```
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
```

```
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();
```

```
        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
```

```
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik

- PutUserPolitik

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
```

```
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
```

```
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
```

```
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret
```

```
 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)
```

```
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Die in diesem Szenario verwendeten IAM-Funktionen.

```
################################################################################
```

```
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
```

```
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
```

```
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
```

```
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
```

```
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
```

```
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
```

```
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
```

```
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    k) access_key="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi
```

```
  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- API-Details finden Sie in den folgenden Themen der AWS CLI -Befehlsreferenz.
  - AttachRoleRichtlinie
  - CreateAccessSchlüssel

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessSchlüssel
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }
```

```
    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }
```

```
        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
```

```
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
```

```
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);
```

```
        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
```

```
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }
```

```
    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
```

```
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
   bool result = true;
   if (policy.ArnHasBeenSet()) {
       // Detach the policy from the role.
       {
           Aws::IAM::Model::DetachRolePolicyRequest request;
           request.SetPolicyArn(policy.GetArn());
           request.SetRoleName(role.GetRoleName());

           Aws::IAM::Model::DetachRolePolicyOutcome outcome =
client.DetachRolePolicy(
                   request);
           if (!outcome.IsSuccess()) {
               std::cerr << "Error Detaching policy from roles. " <<
                         outcome.GetError().GetMessage() << std::endl;
               result = false;
           }
           else {
               std::cout << "Successfully detached the policy with arn "
                         << policy.GetArn()
                         << " from role " << role.GetRoleName() << "." <<
std::endl;
           }
       }

       // Delete the policy.
       {
           Aws::IAM::Model::DeletePolicyRequest request;
           request.WithPolicyArn(policy.GetArn());

           Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
           if (!outcome.IsSuccess()) {
               std::cerr << "Error deleting policy. " <<
                         outcome.GetError().GetMessage() << std::endl;
               result = false;
           }
           else {
               std::cout << "Successfully deleted the policy with arn "
                         << policy.GetArn() << std::endl;
           }
       }
```

```
    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for C++ -API-Referenz.
  - AttachRolePolitik

- CreateAccessSchlüssel
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessSchlüssel
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
// 1. Create a user who has no permissions.
// 2. Create a role that grants permission to list Amazon Simple Storage Service
//    (Amazon S3) buckets for the account.
// 3. Add a policy to let the user assume the role.
// 4. Try and fail to list buckets without permissions.
// 5. Assume the role and list S3 buckets using temporary credentials.
// 6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
```

```
 sdkConfig aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper actions.PolicyWrapper
 roleWrapper actions.RoleWrapper
 userWrapper actions.UserWrapper
 questioner demotools.IQuestioner
 helper IScenarioHelper
 isTestRun bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
  helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:   sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run() {
```

```
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser() *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
```

```
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(user *types.User)
 *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err :=
 scenario.roleWrapper.CreateRole(scenario.helper.GetName(), *user.Arn)
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"}, "arn:aws:s3:::*")
 if err != nil {panic(err)}
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(*listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {panic(err)}
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(*user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {panic(err)}
```

```
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(accessKey
 *types.AccessKey) *aws.Config {
  log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
  scenario.questioner.Ask("Press Enter when you're ready.")
  noPermsConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
 ))
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

  s3Client := s3.NewFromConfig(noPermsConfig)
  _, err = s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
```

```
    panic(err)
   }
  }
  } else {
   log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
    "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
  return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
// 1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//   the user's access key credentials.
// 2. Gets temporary credentials by assuming the role that grants permission to
 list the
//    buckets.
// 3. Creates an Amazon S3 client from the temporary credentials.
// 4. Lists buckets for the account. Because the temporary credentials are
 generated by
//    assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(noPermsConfig
 *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(context.TODO(),
 &sts.AssumeRoleInput{
  RoleArn:           role.Arn,
  RoleSessionName:   aws.String("AssumeRoleExampleSession"),
  DurationSeconds:   aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
```

```
    *tempCredentials.Credentials.SecretAccessKey,
    *tempCredentials.Credentials.SessionToken),
   ),
  )
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
  purposes.
  scenario.addTestOptions(&assumeRoleConfig)

  s3Client := s3.NewFromConfig(assumeRoleConfig)
  result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   log.Println("Couldn't list buckets with assumed role credentials.")
   panic(err)
  }
  log.Println("Successfully called ListBuckets with assumed role credentials, \n"
  +
   "here are some of them:")
  for i := 0; i < len(result.Buckets) && i < 5; i++ {
   log.Printf("\t%v\n", *result.Buckets[i].Name)
  }
  log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(user *types.User, role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
   policies, err := scenario.roleWrapper.ListAttachedRolePolicies(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*policy.PolicyArn)
    if err != nil {panic(err)}
    log.Printf("Detached policy %v from role %v and deleted the policy.\n",
     *policy.PolicyName, *role.RoleName)
   }
   err = scenario.roleWrapper.DeleteRole(*role.RoleName)
   if err != nil {panic(err)}
   log.Printf("Deleted role %v.\n", *role.RoleName)
```

```
   userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
   if err != nil {panic(err)}
   for _, userPol := range userPols {
    err = scenario.userWrapper.DeleteUserPolicy(*user.UserName, userPol)
    if err != nil {panic(err)}
    log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
   }
   keys, err := scenario.userWrapper.ListAccessKeys(*user.UserName)
   if err != nil {panic(err)}
   for _, key := range keys {
    err = scenario.userWrapper.DeleteAccessKey(*user.UserName, *key.AccessKeyId)
    if err != nil {panic(err)}
    log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
   }
   err = scenario.userWrapper.DeleteUser(*user.UserName)
   if err != nil {panic(err)}
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}
```

Definieren Sie eine Struktur, die Kontoaktionen umschließt.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
```

```
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Definieren Sie eine Struktur, die Richtlinienaktionen umschließt.

```


// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
```

```
 Effect string
 Action []string
 Principal map[string]string `json:",omitempty"`
 Resource *string `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
```

```
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

```
// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Definieren Sie eine Struktur, die Rollenaktionen umschließt.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

```
// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
```

```
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

```
// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
```

```
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Definieren Sie eine Struktur, die Benutzeraktionen umschließt.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
```

```
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
```

```
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
```

```
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
```

```
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Go -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel

- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Benutzer-Aktionen umschließen.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */
```

```
public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        String policyName = args[1];
        String roleName = args[2];
```

```
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
```

```
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
```

```
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();
```

```
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void deleteRole(IamClient iam, String roleName, String polArn)
{

       try {
           // First the policy needs to be detached.
           DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                   .policyArn(polArn)
                   .roleName(roleName)
                   .build();

           iam.detachRolePolicy(rolePolicyRequest);

           // Delete the policy.
           DeletePolicyRequest request = DeletePolicyRequest.builder()
                   .policyArn(polArn)
                   .build();

           iam.deletePolicy(request);
           System.out.println("*** Successfully deleted " + polArn);

           // Delete the role.
           DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           iam.deleteRole(roleRequest);
           System.out.println("*** Successfully deleted " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void deleteKey(IamClient iam, String username, String
accessKey) {
       try {
           DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                   .accessKeyId(accessKey)
                   .userName(username)
                   .build();

           iam.deleteAccessKey(request);
```

```
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik

- PutUserPolitik

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer und eine Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen. Nachdem Sie die Rolle übernommen haben, verwenden Sie temporäre Anmeldeinformationen, um Buckets für das Konto aufzulisten.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";
```

```
/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    } else {
      throw caught;
    }
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);
```

```
 if (!User) {
   throw new Error("User not created");
 }

 // Create an access key. This key is used to authenticate the new user to
 // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
 // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
 const createAccessKeyResponse = await iamClient.send(
   new CreateAccessKeyCommand({ UserName: userName }),
 );

 if (
   !createAccessKeyResponse.AccessKey?.AccessKeyId ||
   !createAccessKeyResponse.AccessKey?.SecretAccessKey
 ) {
   throw new Error("Access key not created");
 }

 const {
   AccessKey: { AccessKeyId, SecretAccessKey },
 } = createAccessKeyResponse;

 let s3Client = new S3Client({
   credentials: {
     accessKeyId: AccessKeyId,
     secretAccessKey: SecretAccessKey,
   },
 });

 // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
 // thrown while the user and access keys are still stabilizing.
 await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
   try {
     return await listBuckets(s3Client);
   } catch (err) {
     if (err instanceof Error && err.name === "InvalidAccessKeyId") {
       throw err;
     }
   }
 });
```

```
 // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
 // is thrown while the user and access keys are still stabilizing.
 const { Role } = await retry(
   {
     intervalInMs: 2000,
     maxRetries: 60,
   },
   () =>
     iamClient.send(
       new CreateRoleCommand({
         AssumeRolePolicyDocument: JSON.stringify({
           Version: "2012-10-17",
           Statement: [
             {
               Effect: "Allow",
               Principal: {
                 // Allow the previously created user to assume this role.
                 AWS: User.Arn,
               },
               Action: "sts:AssumeRole",
             },
           ],
         }),
         RoleName: roleName,
       }),
     ),
 );

 if (!Role) {
   throw new Error("Role not created");
 }

 // Create a policy that allows the user to list S3 buckets.
 const { Policy: listBucketPolicy } = await iamClient.send(
   new CreatePolicyCommand({
     PolicyDocument: JSON.stringify({
       Version: "2012-10-17",
       Statement: [
         {
           Effect: "Allow",
           Action: ["s3:ListAllMyBuckets"],
           Resource: "*",
         },
```

```
        ],
      }),
      PolicyName: policyName,
    }),
  );

  if (!listBucketPolicy) {
    throw new Error("Policy not created");
  }

  // Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
  await iamClient.send(
    new AttachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  // Assume the role.
  const stsClient = new STSClient({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the assume role operation until it succeeds.
  const { Credentials } = await retry(
    { intervalInMs: 2000, maxRetries: 60 },
    () =>
      stsClient.send(
        new AssumeRoleCommand({
          RoleArn: Role.Arn,
          RoleSessionName: `iamBasicScenarioSession-${Math.floor(
            Math.random() * 1000000,
          )}`,
          DurationSeconds: 900,
        }),
      ),
  );

  if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
    throw new Error("Credentials not created");
  }
```

```
s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 60 }, () =>
  listBuckets(s3Client),
);

// Clean up.
await iamClient.send(
  new DetachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeletePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
  }),
);

await iamClient.send(
  new DeleteRoleCommand({
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeleteAccessKeyCommand({
    UserName: userName,
    AccessKeyId,
  }),
);

await iamClient.send(
```

```
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for JavaScript -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Benutzer-Aktionen umschließen.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
```

```
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue: String =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
```

```
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
```

```
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String
) {
    val stsClient =
        StsClient {
```

```
            region = "us-east-1"
        }

    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials =
        StaticCredentialsProvider {
            accessKeyId = key
            secretAccessKey = secKey
            sessionToken = secToken
        }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 =
        S3Client {
            credentialsProvider = staticCredentials
            region = "us-east-1"
        }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}
```

```
suspend fun deleteRole(
    roleNameVal: String,
    polArn: String
) {
    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}
```

```
@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace Iam\Basics;
```

```
require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";
```

```
$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}
```

```
$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for PHP -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer und eine Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen. Nachdem Sie die Rolle übernommen haben, verwenden Sie temporäre Anmeldeinformationen, um Buckets für das Konto aufzulisten.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
```

```
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
```

```
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
```

```
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
```

```
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
account.
    Uses the temporary credentials from the role to list the buckets that are
owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
```

```
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser

- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer und eine Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen. Nachdem Sie die Rolle übernommen haben, verwenden Sie temporäre Anmeldeinformationen, um Buckets für das Konto aufzulisten.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts("Give AWS time to propagate resources...")
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
```

```
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Tried and failed to create demo user.")
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
                    Effect: "Allow",
                    Principal: {'AWS': user.arn},
                    Action: "sts:AssumeRole"
                  }]
    }.to_json
```

```
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                    Effect: "Allow",
                    Action: "s3:ListAllMyBuckets",
                    Resource: "arn:aws:s3:::*"
                  }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
```

```
 end

 # Creates an inline policy for a user that lets the user assume a role.
 #
 # @param policy_name [String] The name to give the policy.
 # @param user [Aws::IAM::User] The user that owns the policy.
 # @param role [Aws::IAM::Role] The role that can be assumed.
 # @return [Aws::IAM::UserPolicy] The newly created policy.
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: "2012-10-17",
     Statement: [{
                  Effect: "Allow",
                  Action: "sts:AssumeRole",
                  Resource: role.arn
                }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
```

```
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
     break if count.zero?
   end
 rescue Aws::Errors::ServiceError => e
   if e.code == "AccessDenied"
     puts("Attempt to list buckets with no permissions: AccessDenied.")
   else
     @logger.info("Couldn't list buckets for the account. Here's why: ")
     @logger.info("\t#{e.code}: #{e.message}")
     raise
   end
 end

 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
```

```
      role_session_name: "create-use-assume-role-scenario"
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end

  # Deletes a role. If the role has policies attached, they are detached and
  # deleted before the role is deleted.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Ruby -API-Referenz.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitik
  - DetachRolePolitik
  - PutUserPolitik

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
```

```
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();
```

```
    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
```

```
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
```

```
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
```

```
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zu AWS - SDK für Rust.
  - AttachRolePolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessSchlüssel

- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitik
- DetachRolePolitik
- PutUserPolitik

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen Sie IAM-Benutzer mit Schreibzugriff und Lese-/Schreibzugriff mithilfe eines SDK AWS

Das folgende Codebeispiel veranschaulicht, wie Sie Benutzer erstellen und ihnen Richtlinien zuweisen.

> **Warning**
>
> Um Sicherheitsrisiken zu vermeiden, sollten Sie IAM-Benutzer nicht zur Authentifizierung verwenden, wenn Sie eigens entwickelte Software entwickeln oder mit echten Daten arbeiten. Verwenden Sie stattdessen den Verbund mit einem Identitätsanbieter wie AWS IAM Identity Center.

- Erstellen Sie zwei IAM-Benutzer.
- Fügen Sie eine Richtlinie zu einem Benutzer hinzu und bringen Sie Objekte in einem Amazon S3-Bucket unter.
- Fügen Sie eine Richtlinie für den zweiten Benutzer hinzu, mit der dieser Objekte aus dem Bucket abrufen kann.
- Erhalten Sie unterschiedliche Berechtigungen für den Bucket basierend auf Anmeldeinformationen des Benutzers.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Benutzer-Aktionen umschließen.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user



def update_user(user_name, new_user_name):
```

```
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
```

```
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

Erstellen Sie Funktionen, die IAM-Richtlinien-Aktionen umschließen.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
```

```
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Erstellen Sie Funktionen, die IAM-Zugriffsschlüssel-Aktionen umschließen.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
```

```
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Verwenden Sie die Wrapper-Funktionen, um Benutzer mit unterschiedlichen Richtlinien zu erstellen, und verwenden Sie deren Anmeldeinformationen, um auf einen Amazon-S3-Bucket zuzugreifen.

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
    This demonstration creates two users: one user who can put and get objects in
 an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
 permitted
```

```
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
```

```
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )
```

```
    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
```

```
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - AttachUserPolitik
  - CreateAccessSchlüssel
  - CreatePolicy
  - CreateUser
  - DeleteAccessSchlüssel
  - DeletePolicy
  - DeleteUser
  - DetachUserPolitik
  - ListUsers
  - UpdateUser

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwaltung von IAM-Zugriffsschlüsseln mithilfe eines SDK AWS

Das folgende Codebeispiel veranschaulicht, wie Sie Zugriffsschlüssel verwalten.

> **Warning**
>
> Um Sicherheitsrisiken zu vermeiden, sollten Sie IAM-Benutzer nicht zur Authentifizierung verwenden, wenn Sie eigens entwickelte Software entwickeln oder mit echten Daten arbeiten. Verwenden Sie stattdessen den Verbund mit einem Identitätsanbieter wie AWS IAM Identity Center.

- Erstellen und Listen Sie Zugriffsschlüssel auf.
- Finden Sie heraus, wann und wie ein Zugriffsschlüssel zuletzt verwendet wurde.
- Zugriffsschlüssel aktualisieren und löschen.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Zugriffsschlüssel-Aktionen umschließen.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")
```

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
```

```
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response



def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
```

```
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Verwenden Sie die Wrapper-Funktionen, um Zugriffsschlüssel-Aktionen für den aktuellen Benutzer auszuführen.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
```

```
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
{new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
{last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - CreateAccessSchlüssel
  - DeleteAccessSchlüssel
  - GetAccessKeyLastBenutzt
  - ListAccessSchlüssel
  - UpdateAccessSchlüssel

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwaltung von IAM-Richtlinien mithilfe eines SDK AWS

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen und Auflisten von Richtlinien.
- Erstellen und Abrufen von Richtlinienversionen.
- Zurücksetzen einer Richtlinie auf eine frühere Version.
- Löschen von Richtlinien.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Richtlinien-Aktionen umschließen.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.
```

```
    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
```

```
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version
```

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
```

```
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
 rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
 default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
 so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
```

```
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Verwenden Sie die Wrapper-Funktionen, um Richtlinien zu erstellen, Versionen zu aktualisieren und Informationen über sie zu erhalten.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::made-up-bucket-name"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
```

```
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersión
  - GetPolicyVersión
  - ListPolicies
  - ListPolicyVersionen
  - SetDefaultPolicyVersion

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwaltung von IAM-Rollen mithilfe eines SDK AWS

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen Sie eine IAM-Rolle.
- Anfügen und Trennen von Richtlinien für eine Rolle.
- Löschen Sie eine Rolle.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Rollen-Aktionen umschließen.

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
```

```
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
```

```
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Verwenden Sie die Wrapper-Funktionen, um eine Rolle zu erstellen, und fügen Sie dann eine Richtlinie hinzu und trennen Sie sie.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
 "batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
```

```
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - AttachRolePolitik
  - CreateRole
  - DeleteRole
  - DetachRolePolitik

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwalten Sie Ihr IAM-Konto mit einem SDK AWS

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Rufen Sie den Konto-Alias ab und aktualisieren Sie ihn.
- Generieren Sie einen Bericht über Benutzer und Anmeldeinformationen.
- Rufen Sie eine Zusammenfassung der Kontonutzung ab.
- Rufen Sie Details zu allen Benutzern, Gruppen, Rollen und Richtlinien in Ihrem Konto ab, einschließlich deren Beziehungen untereinander.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die IAM-Kontoaktionen umschließen.

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise



def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
```

```
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
```

```
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Rufen Sie Wrapper-Funktionen auf, um den Konto-Alias zu ändern und Berichte über das Konto zu erhalten.

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
```

```
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
```

```
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialBericht
  - GetAccountAuthorizationDetails
  - GetAccountZusammenfassung
  - GetCredentialBericht
  - ListAccountAliase

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Rollback einer IAM-Richtlinienversion mithilfe eines SDK AWS

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Rufen Sie die Liste der Richtlinienversionen nach Datum sortiert ab.
- Suchen Sie die Standard-Richtlinienversion.
- Machen Sie die vorherige Version der Richtlinie zur Standardversion.
- Löschen Sie die alte Standardversion.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
def rollback_policy_version(policy_arn):
```

```
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
```

```
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - DeletePolicyVersion
  - ListPolicyVersionen
  - SetDefaultPolicyVersion

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Arbeiten Sie mit der IAM Policy Builder-API mithilfe eines SDK AWS

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen Sie IAM-Richtlinien mithilfe der objektorientierten API.
- Verwenden Sie die IAM-Policy-Builder-API mit dem IAM-Service.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In den Beispielen werden folgende Importe verwendet.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Erstellen Sie eine zeitbasierte Richtlinie.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_GREATER_THAN)

 .key("aws:CurrentTime")

 .value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_LESS_THAN)

 .key("aws:CurrentTime")
```

```
 .value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
 readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

Erstellen Sie eine Richtlinie mit mehreren Bedingungen.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

 .addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

 .addAction("dynamodb:BatchWriteItem")

 .addResource("arn:aws:dynamodb:*:*:table/table-name")

 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                        List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))
```

```
.key("dynamodb:Select")

.value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Verwenden Sie Prinzipale in einer Richtlinie.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)

.addResource("arn:aws:s3:::BUCKETNAME/*")

.addResource("arn:aws:s3:::BUCKETNAME")
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.ARN_NOT_EQUALS)

.key("aws:PrincipalArn")

.value("arn:aws:iam::444455556666:user/user-name")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Gewähren Sie kontoübergreifenden -Zugriff.

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
```

```
 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::DOC-
EXAMPLE-BUCKET/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                    .key("s3:x-amz-
acl")
                                                                    .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Erstellen und laden Sie eine `IamPolicy` hoch.

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

Downloaden und arbeiten Sie mit einer `IamPolicy`.

```
       public String createNewBasedOnExistingPolicyExample(IamClient iam, String
accountID, String policyName,
                       String newPolicyName) {

               String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
policyName;
               GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
r.policyArn(policyArn));

               String policyVersion =
getPolicyResponse.policy().defaultVersionId();
               GetPolicyVersionResponse getPolicyVersionResponse = iam
                               .getPolicyVersion(r ->
r.policyArn(policyArn).versionId(policyVersion));

               // Create an IamPolicy instance from the JSON string returned
from IAM.
               String decodedPolicy =
URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                               StandardCharsets.UTF_8);
               IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

               /*
                * All IamPolicy components are immutable, so use the copy method
that creates a
                * new instance that
                * can be altered in the same method call.
                *
                * Add the ability to get an item from DynamoDB as an additional
action.
                */
               IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

               // Create a new statement that replaces the original statement.
               IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

               // Upload the new policy. IAM now has both policies.
               iam.createPolicy(r -> r.policyName(newPolicyName)
                               .policyDocument(newPolicy.toJson()));
```

```
                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Weitere Informationen finden Sie im AWS SDK for Java 2.x -Entwicklerhandbuch.
- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele für die AWS STS Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Verwendung AWS STS mit einem AWS Software Development Kit (SDK) funktioniert.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie eine bestimmte Aufgabe ausführen können, indem Sie mehrere Funktionen innerhalb desselben Services aufrufen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Aktionen für die AWS STS Verwendung von AWS SDKs
  - Verwendung AssumeRole mit einem AWS SDK oder CLI
  - Verwendung AssumeRoleWithWebIdentity mit einem AWS SDK oder CLI
  - Verwendung DecodeAuthorizationMessage mit einem AWS SDK oder CLI

Aktionen für die AWS STS Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne AWS STS Aktionen mit AWS SDKs ausgeführt werden. Diese Auszüge rufen die AWS STS API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie unter AWS Security Token Service (AWS STS)-API-Referenz.

Beispiele

- Verwendung AssumeRole mit einem AWS SDK oder CLI
- Verwendung AssumeRoleWithWebIdentity mit einem AWS SDK oder CLI
- Verwendung DecodeAuthorizationMessage mit einem AWS SDK oder CLI
- Verwendung GetFederationToken mit einem AWS SDK oder CLI
- Verwendung GetSessionToken mit einem AWS SDK oder CLI

Verwendung **`AssumeRole`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AssumeRole`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Übernehmen Sie eine IAM-Rolle, die ein MFA-Token erfordert
- Erstellen einer URL mit für Verbundbenutzer

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
```

```
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for .NET API-Referenz.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
```

```
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS CLI Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So übernehmen Sie eine Rolle

Der folgende `assume-role`-Befehl ruft eine Reihe von kurzfristigen Anmeldeinformationen für die IAM-Rolle `s3-access-example` ab.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Ausgabe:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
```

```
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

Die Ausgabe des Befehls enthält einen Zugriffsschlüssel, einen geheimen Schlüssel und ein Sitzungs-Token, die Sie zur Authentifizierung bei AWS verwenden können.

Für die Verwendung AWS über die CLI können Sie ein benanntes Profil einrichten, das einer Rolle zugeordnet ist. Wenn Sie das Profil verwenden, ruft die AWS CLI assume-role auf und verwaltet die Anmeldeinformationen für Sie. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle in der AWS CLI im AWS CLI-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AssumeRole AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
```

```
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
```

```
                    roleSessionName - An identifier for the assumed role session
(for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
exTime);
```

```
        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie den Client.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Übernehmen Sie die IAM-Rolle.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
```

```
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });
```

```
//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for JavaScript API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Gibt einen Satz temporärer Anmeldeinformationen (Zugriffsschlüssel, geheimer Schlüssel und Sitzungstoken) zurück, die eine Stunde lang für den Zugriff auf AWS Ressourcen verwendet werden können, auf die der anfragende Benutzer normalerweise keinen Zugriff hat. Die zurückgegebenen Anmeldeinformationen verfügen über die Berechtigungen, die durch die Zugriffsrichtlinie der übernommenen Rolle und die angegebene Richtlinie zulässig sind (Sie können die bereitgestellte Richtlinie nicht verwenden, um

Berechtigungen zu gewähren, die über die in der Zugriffsrichtlinie der übernommenen Rolle definierten Berechtigungen hinausgehen).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Beispiel 2: Gibt einen Satz temporärer Anmeldeinformationen zurück, die für eine Stunde gültig sind und dieselben Berechtigungen haben, die in der Zugriffsrichtlinie der Rolle definiert sind, die angenommen wird.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Beispiel 3: Gibt einen Satz temporärer Anmeldeinformationen zurück, die die Seriennummer und das generierte Token aus einem MFA enthalten, das den Benutzeranmeldedaten zugeordnet ist, die zur Ausführung des Cmdlets verwendet wurden.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Beispiel 4: Gibt einen Satz temporärer Anmeldeinformationen zurück, die eine in einem Kundenkonto definierte Rolle übernommen haben. Für jede Rolle, die der Drittanbieter übernehmen kann, muss das Kundenkonto eine Rolle mithilfe einer Kennung erstellen, die bei jeder Übernahme der Rolle im ExternalId Parameter - übergeben werden muss.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Einzelheiten zur API finden Sie unter AssumeRole AWS Tools for PowerShellCmdlet-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Übernehmen Sie eine IAM-Rolle, die ein MFA-Token erfordert, und verwenden Sie temporäre Anmeldeinformationen, um Amazon-S3-Buckets für das Konto aufzulisten.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")
```

```
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Einzelheiten zur API finden Sie AssumeRolein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
```

```
  # @param role_arn [String] The ARN of the role that is assumed when these
 credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
 assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: "create-use-assume-role-scenario"
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

- Einzelheiten zur API finden Sie AssumeRolein der AWS SDK for Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
```

```
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Einzelheiten zur API finden Sie AssumeRolein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
                    async throws -> STSClientTypes.Credentials {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
```

```
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            throw error
        }
    }
```

- Einzelheiten zur API finden Sie AssumeRolein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von IAM mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AssumeRoleWithWebIdentity`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AssumeRoleWithWebIdentity`.

CLI

AWS CLI

Um kurzfristige Anmeldeinformationen für eine Rolle zu erhalten, die mit Web Identity (OAuth 2.0") authentifiziert wurde

Der folgende `assume-role-with-web-identity`-Befehl ruft eine Reihe von kurzfristigen Anmeldeinformationen für die IAM-Rolle `app1` ab. Die Anfrage wird mithilfe des Webidentitäts-Tokens authentifiziert, das vom angegebenen Web-Identitätsanbieter bereitgestellt wird. Zwei zusätzliche Richtlinien werden auf die Sitzung angewendet, um die Möglichkeiten des Benutzers weiter einzuschränken. Die zurückgegebenen Anmeldeinformationen laufen eine Stunde nach ihrer Generierung ab.

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
```

```
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

Ausgabe:

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A"
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    }
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

Weitere Informationen finden Sie unter Anfordern von temporären Sicherheitsanmeldeinformationen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter AssumeRoleWithWebIdentität in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Gibt einen temporären Satz von Anmeldeinformationen zurück, der für eine Stunde gültig ist, für einen Benutzer, der mit dem Identity-Provider „Login with Amazon“ authentifiziert wurde. Die Anmeldeinformationen gehen von der Zugriffsrichtlinie aus, die mit der Rolle verknüpft ist, die durch den Rollen-ARN identifiziert wurde. Optional können Sie dem Parameter -Policy eine JSON-Richtlinie übergeben, die die Zugriffsberechtigungen weiter verfeinert (Sie können nicht mehr Berechtigungen gewähren, als in den mit der Rolle verknüpften Berechtigungen verfügbar sind). Der an - übergebene Wert WebIdentityToken ist die eindeutige Benutzer-ID, die vom Identitätsanbieter zurückgegeben wurde.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Einzelheiten zur API finden Sie unter Referenz zur AssumeRoleWithWebIdentität im AWS Tools for PowerShell Cmdlet.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DecodeAuthorizationMessage`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DecodeAuthorizationMessage`.

CLI

AWS CLI

Um eine codierte Autorisierungsnachricht zu dekodieren, die als Antwort auf eine Anfrage zurückgegeben wird

Das folgende `decode-authorization-message` Beispiel dekodiert zusätzliche Informationen über den Autorisierungsstatus einer Anfrage aus einer codierten Nachricht, die als Antwort auf eine Amazon Web Services Services-Anfrage zurückgegeben wurde.

```
aws sts decode-authorization-message \
```

```
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

Ausgabe:

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```

Weitere Informationen finden Sie unter Bewertungslogik für Richtlinien im AWS IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter DecodeAuthorizationMessage in AWS CLI Command Reference.

PowerShell

Tools für PowerShell

Beispiel 1: Dekodiert die zusätzlichen Informationen, die im bereitgestellten codierten Nachrichteninhalt enthalten sind und als Antwort auf eine Anfrage zurückgegeben wurden. Die zusätzlichen Informationen sind verschlüsselt, da es sich bei Details zum Autorisierungsstatus um vertrauliche Informationen handeln kann, die der Benutzer, der die Aktion angefordert hat, nicht sehen sollte.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Einzelheiten zur API finden Sie unter DecodeAuthorizationMessage in AWS Tools for PowerShell Cmdlet Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetFederationToken`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetFederationToken`.

CLI

AWS CLI

Um einen Satz temporärer Sicherheitsanmeldedaten mithilfe von IAM-Benutzerzugriffsschlüsselanmeldedaten zurückzugeben

Im folgenden `get-federation-token` Beispiel wird ein Satz temporärer Sicherheitsanmeldeinformationen (bestehend aus einer Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sicherheitstoken) für einen Benutzer zurückgegeben. Sie müssen

den `GetFederationToken` Vorgang mit den langfristigen Sicherheitsanmeldeinformationen eines IAM-Benutzers aufrufen.

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

Inhalt von `myfile.json`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

Ausgabe:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
    "PackedPolicySize": 36
}
```

Weitere Informationen finden Sie unter Anfordern von temporären Sicherheitsanmeldeinformationen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetFederationToken in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Fordert ein Verbundtoken an, das für eine Stunde gültig ist, wobei „Bob“ als Name des Verbundbenutzers verwendet wird. Dieser Name kann verwendet werden, um in einer ressourcenbasierten Richtlinie (z. B. einer Amazon S3 S3-Bucket-Richtlinie) auf den Verbundbenutzernamen zu verweisen. Die bereitgestellte IAM-Richtlinie im JSON-Format wird verwendet, um den Umfang der Berechtigungen einzuschränken, die dem IAM-Benutzer zur Verfügung stehen. Die bereitgestellte Richtlinie kann nicht mehr Berechtigungen gewähren als die, die dem anfragenden Benutzer gewährt werden, wobei die endgültigen Berechtigungen für den Verbundbenutzer aufgrund der Schnittmenge zwischen der übergebenen Richtlinie und der IAM-Benutzerrichtlinie am restriktivsten sind.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Einzelheiten zur API finden Sie unter Referenz zu GetFederationToken in AWS Tools for PowerShell Cmdlets.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetSessionToken`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetSessionToken`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Abrufen eines Sitzungs-Tokens, das ein MFA-Token erfordert

CLI

AWS CLI

So erhalten Sie einen Satz kurzfristiger Anmeldeinformationen für eine IAM-Identität

Der folgende `get-session-token`-Befehl ruft einen Satz kurzfristiger Anmeldeinformationen für die IAM-Identität ab, die den Aufruf ausführt. Die resultierenden Anmeldeinformationen können für Anfragen verwendet werden, bei denen die Richtlinie eine Multi-Faktor-Authentifizierung (MFA) erfordert. Die Anmeldeinformationen verfallen 15 Minuten nach ihrer Generierung.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Ausgabe:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
```

```
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Weitere Informationen finden Sie unter Anfordern von temporären Sicherheitsanmeldeinformationen im AWS -IAM-Benutzerhandbuch.

- Einzelheiten zur API finden Sie unter GetSessionToken in der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Gibt eine **`Amazon.RuntimeAWSCredentials`** Instanz zurück, die temporäre Anmeldeinformationen enthält, die für einen bestimmten Zeitraum gültig sind. Die Anmeldeinformationen, die zum Anfordern temporärer Anmeldeinformationen verwendet werden, werden aus den aktuellen Shell-Standardeinstellungen abgeleitet. Um andere Anmeldeinformationen anzugeben, verwenden Sie die Parameter - ProfileName oder -AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Ausgabe:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Beispiel 2: Gibt eine **`Amazon.RuntimeAWSCredentials`** Instanz zurück, die temporäre Anmeldeinformationen enthält, die für eine Stunde gültig sind. Die für die Anfrage verwendeten Anmeldeinformationen stammen aus dem angegebenen Profil.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Ausgabe:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Beispiel 3: Gibt eine **`Amazon.RuntimeAWSCredentials`** Instanz mit temporären Anmeldeinformationen zurück, die für eine Stunde gültig sind. Dabei werden die Identifikationsnummer des MFA-Geräts, das dem Konto zugeordnet ist, dessen Anmeldeinformationen im Profil „myprofilename“ angegeben sind, und der vom Gerät bereitgestellte Wert verwendet.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Ausgabe:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Einzelheiten zur API finden Sie unter Referenz zu GetSessionToken in AWS Tools for PowerShell Cmdlets.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie ein Sitzungs-Token ab, indem Sie ein MFA-Token übergeben, und verwenden Sie es, um Amazon-S3-Buckets für das Konto aufzulisten.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Einzelheiten zur API finden Sie unter API-Referenz zu GetSessionToken in AWS SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für die AWS STS Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen Ihnen, wie Sie allgemeine Szenarien AWS STS mit AWS SDKs implementieren. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben erledigen können, indem Sie darin mehrere Funktionen aufrufen. AWS STS Jedes Szenario enthält einen Link zu GitHub, über den Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Beispiele

- Gehen Sie von einer IAM-Rolle aus, für die ein MFA-Token erforderlich ist, und AWS STS verwenden Sie ein SDK AWS
- Konstruieren Sie mithilfe eines SDK eine AWS URL AWS STS für Verbundbenutzer
- Rufen Sie AWS STS mithilfe eines SDK ein Sitzungstoken ab, für das ein MFA-Token erforderlich ist AWS

Gehen Sie von einer IAM-Rolle aus, für die ein MFA-Token erforderlich ist, und AWS STS verwenden Sie ein SDK AWS

Das folgende Code-Beispiel veranschaulicht, wie Sie eine Rolle anfügen, die ein MFA-Token erfordert.

> **Warning**
>
> Um Sicherheitsrisiken zu vermeiden, sollten Sie IAM-Benutzer nicht zur Authentifizierung verwenden, wenn Sie eigens entwickelte Software entwickeln oder mit echten Daten arbeiten. Verwenden Sie stattdessen den Verbund mit einem Identitätsanbieter wie AWS IAM Identity Center.

- Erstellen Sie eine IAM-Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt.
- Erstellen Sie einen IAM-Benutzer, der nur dann berechtigt ist, die Rolle zu übernehmen, wenn MFA-Anmeldeinformationen bereitgestellt werden.
- Registrieren Sie ein MFA-Gerät für den Benutzer.

- Übernehmen der Rolle und Verwendung von temporären Anmeldeinformationen zum Auflisten von Amazon-S3-Buckets.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer, registrieren Sie ein MFA-Gerät, und erstellen Sie eine Rolle, die die Berechtigung zum Auflisten von S3-Buckets gewährt. Der Benutzer hat nur Rechte, um die Rolle anzunehmen.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
 MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    For demonstration purposes, the user is created in the same account as the
 role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
```

```
                        that has permissions to create users, roles, and
policies
                        in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
```

```
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
```

```
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Zeigen Sie, dass die Annahme der Rolle ohne MFA-Token nicht zulässig ist.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

Übernehmen Sie die Rolle, die die Berechtigung zum Auflisten von S3-Buckets erteilt, indem Sie das erforderliche MFA-Token übergeben, und zeigen Sie, dass Buckets aufgelistet werden können.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
```

```
        print(bucket.name)
```

Zerstören Sie die für das Demo erstellten Ressourcen.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Führen Sie dieses Szenario aus, indem Sie die zuvor definierten Funktionen verwenden.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
```

```
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Einzelheiten zur API finden Sie AssumeRolein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Konstruieren Sie mithilfe eines SDK eine AWS URL AWS STS für Verbundbenutzer

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen Sie eine IAM-Rolle, die nur Lesezugriff auf die Amazon-S3-Ressourcen des aktuellen Kontos gewährt.
- Holen Sie sich ein Sicherheitstoken vom AWS Verbundendpunkt.

- Erstellen Sie eine URL, die für den Zugriff auf die Konsole mit Verbund-Anmeldeinformationen verwendet werden kann.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine Rolle, die nur Lesezugriff auf die S3-Ressourcen des aktuellen Kontos gewährt.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
```

```
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return role
```

Rufen Sie ein Sicherheitstoken vom AWS Verbundendpunkt ab und erstellen Sie eine URL, die für den Zugriff auf die Konsole mit Verbundanmeldedaten verwendet werden kann.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
       AWS federation endpoint.
    3. Builds a URL that can be used in a browser to navigate to the AWS
 federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.

    :param assume_role_arn: The role that specifies the permissions that are
 granted.
                            The current user must have permission to assume the
 role.
    :param session_name: The name for the STS session.
    :param issuer: The organization that issues the URL.
    :param sts_client: A Boto3 STS instance that can assume the role.
    :return: The federated URL.
    """
    response = sts_client.assume_role(
```

```
        RoleArn=assume_role_arn, RoleSessionName=session_name
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    session_data = {
        "sessionId": temp_credentials["AccessKeyId"],
        "sessionKey": temp_credentials["SecretAccessKey"],
        "sessionToken": temp_credentials["SessionToken"],
    }
    aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

    # Make a request to the AWS federation endpoint to get a sign-in token.
    # The requests.get function URL-encodes the parameters and builds the query
string
    # before making the request.
    response = requests.get(
        aws_federated_signin_endpoint,
        params={
            "Action": "getSigninToken",
            "SessionDuration": str(datetime.timedelta(hours=12).seconds),
            "Session": json.dumps(session_data),
        },
    )
    signin_token = json.loads(response.text)
    print(f"Got a sign-in token from the AWS sign-in federation endpoint.")

    # Make a federated URL that can be used to sign into the AWS Management
Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Zerstören Sie die für das Demo erstellten Ressourcen.

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Führen Sie dieses Szenario aus, indem Sie die zuvor definierten Funktionen verwenden.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
```

```
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Einzelheiten zur API finden Sie AssumeRolein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Rufen Sie AWS STS mithilfe eines SDK ein Sitzungstoken ab, für das ein MFA-Token erforderlich ist AWS

Das folgende Code-Beispiel veranschaulicht, wie Sie ein Sitzungstoken erhalten, das ein MFA-Token erfordert.

> **Warning**
>
> Um Sicherheitsrisiken zu vermeiden, sollten Sie IAM-Benutzer nicht zur Authentifizierung verwenden, wenn Sie eigens entwickelte Software entwickeln oder mit echten Daten arbeiten. Verwenden Sie stattdessen den Verbund mit einem Identitätsanbieter wie AWS IAM Identity Center.

- Erstellen Sie eine IAM-Rolle, die die Berechtigung zum Auflisten von Amazon-S3-Buckets erteilt.
- Erstellen Sie einen IAM-Benutzer, der nur dann berechtigt ist, die Rolle zu übernehmen, wenn MFA-Anmeldeinformationen bereitgestellt werden.
- Registrieren Sie ein MFA-Gerät für den Benutzer.
- Stellen Sie MFA-Anmeldeinformationen bereit, um ein Sitzungstoken abzurufen, und verwenden Sie temporäre Anmeldeinformationen, um S3-Buckets aufzulisten.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen IAM-Benutzer, registrieren Sie ein MFA-Gerät und erstellen Sie eine Rolle, die dem Benutzer die Berechtigung erteilt, S3-Buckets nur aufzulisten, wenn MFA-Anmeldeinformationen verwendet werden.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
```

```
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
```

```
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Rufen Sie temporäre Sitzungs-Anmeldeinformationen ab, indem Sie ein MFA-Token übergeben, und verwenden Sie die Anmeldeinformationen, um S3-Buckets für das Konto aufzulisten.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
```

```
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Zerstören Sie die für das Demo erstellten Ressourcen.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Führen Sie dieses Szenario aus, indem Sie die zuvor definierten Funktionen verwenden.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
```

```
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Einzelheiten zur API finden Sie unter API-Referenz zu GetSessionToken in AWS SDK for Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden von IAM mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Sicherheit in IAM und AWS STS

Cloud-Sicherheit hat AWS höchste Priorität. Als AWS Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame AWS Verantwortung von Ihnen und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud selbst 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. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS . Weitere Informationen zu den Compliance-Programmen, die für AWS Identity and Access Management (IAM) gelten, finden Sie unter AWS Services im Bereich nach Compliance-Programm AWS .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, einschließlich der Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von AWS Identity and Access Management (IAM) und AWS Security Token Service (AWS STS) anwenden. In den folgenden Themen erfahren Sie, wie Sie IAM konfigurieren und AWS STS Ihre Sicherheits- und Compliance-Ziele erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, die Sie bei der Überwachung und Sicherung Ihrer IAM-Ressourcen unterstützen.

Inhalt

- AWS verwaltete Richtlinien für AWS Identity and Access Management Access Analyzer

AWS Sicherheitsnachweise

Bei der Interaktion mit geben Sie Ihre AWS Sicherheitsanmeldedaten an AWS, um zu überprüfen, wer Sie sind und ob Sie berechtigt sind, auf die von Ihnen angeforderten Ressourcen zuzugreifen. AWS verwendet die Sicherheitsanmeldedaten, um Ihre Anfragen zu authentifizieren und zu autorisieren.

Wenn Sie z. B. eine geschützte Datei aus einem Amazon Simple Storage Service (Amazon S3)-Bucket herunterladen möchten, müssen Ihre Anmeldeinformationen diesen Zugriff erlauben. Wenn aus Ihren Anmeldeinformationen nicht hervorgeht, dass Sie zum Herunterladen der Datei berechtigt sind, AWS lehnt Ihre Anfrage ab. Ihre AWS Sicherheitsanmeldedaten sind jedoch nicht erforderlich, um eine Datei in einem Amazon S3 S3-Bucket herunterzuladen, der öffentlich geteilt wird.

Es gibt verschiedene Benutzertypen in AWS. Alle AWS Benutzer haben Sicherheitsanmeldeinformationen. Es gibt den Kontoinhaber (Root-Benutzer) Benutzer im AWS IAM Identity Center, Verbundbenutzer und IAM-Benutzer.

Benutzer verfügen entweder über langfristige oder temporäre Sicherheitsanmeldeinformationen. Root-Benutzer, IAM-Benutzer und Zugriffsschlüssel haben langfristige Sicherheitsanmeldeinformationen, die nicht ablaufen. Um langfristige Anmeldeinformationen zu schützen, müssen Prozesse zur Verwaltung von Zugriffsschlüsseln, zum Ändern von Passwörtern und zum Aktivieren von MFA vorhanden sein.

IAM-Rollen und Verbundbenutzer verfügen über temporäre Sicherheitsanmeldedaten. Benutzer im AWS IAM Identity Center Temporäre Sicherheitsanmeldeinformationen laufen nach einem definierten Zeitraum ab oder wenn der Benutzer seine Sitzung beendet. Temporäre Anmeldeinformationen funktionieren fast genauso wie die langfristigen Anmeldeinformationen, mit folgenden Unterschieden:

- Temporäre Sicherheitsanmeldeinformationen sind über einen kurzen Zeitraum gültig, wie der Name schon sagt. Sie können mit einer Gültigkeitsdauer von wenigen Minuten bis mehrere Stunden konfiguriert werden. Wenn die Anmeldeinformationen abgelaufen sind, werden sie nicht AWS mehr erkannt und es ist auch kein Zugriff mehr über API-Anfragen möglich, die mit ihnen gestellt wurden.
- Temporäre Sicherheitsanmeldeinformationen werden nicht mit dem Benutzer gespeichert, sondern auf Anforderung des Benutzers dynamisch generiert und bereitgestellt. Wenn (oder sogar bevor) die temporären Anmeldeinformationen ablaufen, kann der Benutzer neue Anmeldeinformationen anfordern, solange der anfordernde Benutzer weiterhin dazu berechtigt ist.

Daher haben temporäre Anmeldeinformationen die folgenden Vorteile gegenüber langfristigen Anmeldeinformationen:

- Sie müssen keine langfristigen AWS Sicherheitsnachweise verteilen oder in eine Anwendung einbetten.
- Sie können Benutzern Zugriff auf Ihre AWS Ressourcen gewähren, ohne eine AWS Identität für sie definieren zu müssen. Temporäre Anmeldeinformationen sind die Grundlage für Rollen und den Identitätsverbund.
- Die temporären Anmeldeinformationen haben eine begrenzte Nutzungsdauer. Somit müssen Sie sie aktualisieren oder explizit widerrufen, wenn Sie sie nicht mehr benötigen. Nachdem die temporären Anmeldeinformationen abgelaufen sind, können sie nicht erneut verwendet werden. Sie können die Gültigkeit der Anmeldeinformationen bis zu einem bestimmten Höchstwert festlegen.

Sicherheitsüberlegungen

Wir empfehlen, die folgenden Informationen zu berücksichtigen, wenn Sie die Datenschutzvorkehrungen für Ihre AWS-Konto festlegen:

- Wenn Sie einen erstellen AWS-Konto, erstellen wir das Konto Root-Benutzer. Diese Anmeldeinformationen des Root-Benutzers (Kontoinhabers) ermöglichen vollständigen Zugriff auf alle Ressourcen des Kontos. Die erste Aufgabe, die Sie mit dem Root-Benutzer ausführen, besteht darin, einem anderen Benutzer Administratorrechte zu gewähren, AWS-Konto sodass Sie die Nutzung des Root-Benutzers minimieren.
- Sie können keine IAM-Richtlinien verwenden, um dem Root-Benutzer den Zugriff auf Ressourcen explizit zu verweigern. Sie können nur eine AWS Organizations Service Control Policy (SCP) verwenden, um die Rechte des Root-Benutzers einzuschränken.
- Wenn Sie Ihr Root-Benutzer-Passwort vergessen oder verlieren, müssen Sie Zugriff auf die mit Ihrem Konto verknüpfte E-Mail-Adresse haben, um es zurücksetzen zu können.
- Wenn Sie Ihre Root-Benutzer-Zugangsschlüssel verlieren, müssen Sie sich in Ihrem Konto als Root-Benutzer anmelden können, um neue zu erstellen.
- Verwenden Sie Ihren Root-Benutzer nicht für alltägliche Aufgaben. Verwenden Sie ihn nur, 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 Stammbenutzer-Anmeldeinformationen erfordern.

- Sicherheitsanmeldeinformationen gelten jeweils für ein Konto. Wenn Sie Zugriff auf mehrere AWS-Konten haben, verfügen Sie über separate Anmeldeinformationen für jedes Konto.
- Richtlinien legen fest, welche Aktionen ein Benutzer, eine Rolle oder ein Mitglied einer Benutzergruppe auf welchen AWS Ressourcen und unter welchen Bedingungen ausführen kann. Mithilfe von Richtlinien können Sie den Zugriff auf AWS-Services und die Ressourcen in Ihrem sicher kontrollieren AWS-Konto. Wenn Sie Berechtigungen als Reaktion auf ein Sicherheitsereignis ändern oder widerrufen müssen, löschen oder ändern Sie die Richtlinien, anstatt Änderungen direkt an der Identität vorzunehmen.
- Stellen Sie sicher, dass Sie die Anmeldeinformationen für Ihren Emergency-Access-IAM-Benutzer und alle Zugriffsschlüssel, die Sie für den programmatischen Zugriff erstellt haben, an einem sicheren Ort speichern. Wenn Sie Ihre Zugriffsschlüssel verlieren, müssen Sie sich in Ihrem Konto anmelden, um neue zu erstellen.
- Es wird dringend empfohlen, temporäre Anmeldeinformationen zu verwenden, die von IAM-Rollen und Verbundbenutzern bereitgestellt werden, anstelle der langfristigen Anmeldeinformationen, die von IAM-Benutzern und Zugriffsschlüsseln bereitgestellt werden.

Verbundidentität

Föderierte Identitäten sind Benutzer mit externen Identitäten, denen temporäre AWS Anmeldeinformationen gewährt werden, mit denen sie auf sichere Ressourcen zugreifen können. AWS-Konto Externe Identitäten können vom Identitätsspeicher eines Unternehmens (z. B. LDAP oder Windows Active Directory) oder von einem Drittanbieter (z. B. Login bei Amazon, Facebook oder Google) stammen. Föderierte Identitäten melden sich nicht beim Portal an oder greifen nicht auf das Portal zu. AWS Management Console AWS

Um Verbundidentitäten die Anmeldung bei AWS zu ermöglichen, müssen Sie eine benutzerdefinierte URL erstellen, die `https://signin.aws.amazon.com/federation` enthält. Weitere Informationen finden Sie unter Aktivieren des benutzerdefinierten Identity Broker-Zugriffs auf die AWS Konsole.

Weitere Informationen zu Verbundidentitäten finden Sie unter Identitätsanbieter und Verbund.

Multi-Faktor-Authentifizierung (MFA)

Die Multi-Faktor-Authentifizierung (MFA) bietet eine zusätzliche Sicherheitsebene für Benutzer, die auf Ihr AWS-Konto zugreifen können. Für zusätzliche Sicherheit empfehlen wir, dass Sie MFA für die

Root-Benutzer des AWS-Kontos -Anmeldeinformationen und alle IAM-Benutzer anfordern. Weitere Informationen finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

Wenn Sie MFA aktivieren und sich bei Ihrem anmelden AWS-Konto, werden Sie aufgefordert, Ihre Anmeldeinformationen sowie eine von einem MFA-Gerät generierte Antwort einzugeben, z. B. einen Code, eine Berührung oder ein biometrischer Scan. Wenn Sie MFA hinzufügen, sind Ihre AWS-Konto Einstellungen und Ressourcen sicherer.

Standardmäßig ist MFA nicht aktiviert. Sie können MFA-Geräte für den Root-Benutzer des AWS-Kontos aktivieren und verwalten, indem Sie die Seite Sicherheitsanmeldeinformationen oder das IAM-Dashboard in der AWS Management Console aufrufen. Weitere Informationen zum Aktivieren der MFA für IAM-Benutzer finden Sie unter Aktivierung von MFA-Geräten für Benutzer in AWS.

Weitere Informationen zur Anmeldung mithilfe von Multifaktor-Authentifizierung-(MFA)-Geräten finden Sie unter Verwenden von MFA-Geräten auf Ihrer IAM-Anmeldeseite.

Programmgesteuerter Zugriff

Sie geben Ihre AWS Zugriffstasten an, um programmgesteuerte Aufrufe zu tätigen AWS oder das AWS Command Line Interface Oder zu verwenden. AWS Tools for PowerShell Wir empfehlen, möglichst temporäre Zugriffsschlüssel zu verwenden.

Wenn Sie einen langfristigen Zugriffsschlüssel erstellen, erstellen Sie die Zugriffsschlüssel-ID (z. B. `AKIAIOSFODNN7EXAMPLE`) und den geheimen Zugriffsschlüssel (z. B. `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) als Set. Der geheime Zugriffsschlüssel ist nur zum Zeitpunkt seiner Erstellung zum Download verfügbar. Wenn Sie den geheimen Zugriffsschlüssel nicht herunterladen oder ihn verlieren, müssen Sie einen neuen erstellen.

In vielen Fällen benötigen Sie keine langfristigen Zugriffsschlüssel, die nie ablaufen (wie es bei IAM-Benutzern der Fall ist, wenn Sie Zugriffsschlüssel erstellen). Erstellen Sie stattdessen IAM-Rollen und generieren Sie temporäre Sicherheitsanmeldeinformationen. Temporäre Sicherheitsanmeldeinformationen enthalten eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel, aber sie enthalten auch ein Sicherheitstoken, das angibt, wann die Anmeldeinformationen ablaufen. Nachdem sie abgelaufen sind, sind sie nicht mehr gültig.

Bei Zugriffsschlüssel-IDs, die mit beginnen, `AKIA` handelt es sich um langfristige Zugriffsschlüssel für einen IAM-Benutzer oder einen AWS-Konto Root-Benutzer. Zugriffsschlüssel-IDs, die mit beginnen, `ASIA` sind temporäre Zugangsschlüssel für Anmeldeinformationen, die Sie mithilfe von AWS STS Vorgängen erstellen.

Benutzer benötigen programmgesteuerten Zugriff, wenn sie mit AWS außerhalb von interagieren möchten. AWS Management Console Die Art und Weise, wie programmatischer Zugriff gewährt wird, hängt vom Benutzertyp ab, der zugreift. AWS

Um Benutzern programmgesteuerten Zugriff zu gewähren, wählen Sie eine der folgenden Optionen.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
Mitarbeiteridentität (Benutzer, die in IAM Identity Center verwaltet werden)	Verwenden Sie temporäre Anmeldeinformationen, um programmatische Anfragen an die AWS CLI, AWS SDKs oder APIs zu signieren. AWS	Befolgen Sie die Anweisungen für die Schnittstelle, die Sie verwenden möchten. • Informationen zu den AWS CLI finden Sie unter Konfiguration der AWS CLI zu AWS IAM Identity Center verwendenden im AWS Command Line Interface Benutzerhandbuch. • Informationen zu AWS SDKs, Tools und AWS APIs finden Sie unter IAM Identity Center-Authentifizierung im Referenzhandbuch für AWS SDKs und Tools.
IAM	Verwenden Sie temporäre Anmeldeinformationen, um programmatische Anfragen an die AWS CLI, AWS SDKs oder APIs zu signieren. AWS	Folgen Sie den Anweisungen unter Verwenden temporärer Anmeldeinformationen mit AWS Ressourcen im IAM-Benutzerhandbuch.
IAM	(Nicht empfohlen) Verwenden Sie langfristige Anmeldeinformationen, um programmatische Anfragen an	Befolgen Sie die Anweisungen für die Schnittstelle, die Sie verwenden möchten.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
	die AWS CLI, AWS SDKs oder APIs zu signieren. AWS	• Informationen dazu finden Sie unter Authentifizierung mithilfe von IAM-Benutzeranmeldedaten im Benutzerhandbuch. AWS CLIAWS Command Line Interface • Informationen zu AWS SDKs und Tools finden Sie unter Authentifizieren mit langfristigen Anmeldeinformationen im Referenzhandbuch für AWS SDKs und Tools. • Informationen zu AWS APIs finden Sie unter Verwaltung von Zugriffsschlüsseln für IAM-Benutzer im IAM-Benutzerhandbuch.

Alternativen zu Langzeit-Zugriffsschlüsseln

Für viele gängige Anwendungsfälle gibt es Alternativen zu langfristigen Zugriffsschlüsseln. Beachten Sie Folgendes, um die Sicherheit Ihres Kontos zu verbessern.

- Betten Sie langfristige Zugriffsschlüssel und geheime Zugriffsschlüssel nicht in Ihren Anwendungscode oder in ein Code-Repository ein. Verwenden AWS Secrets Manager Sie stattdessen eine andere Lösung zur Verwaltung geheimer Geheimnisse, sodass Sie Schlüssel nicht im Klartext fest codieren müssen. Die Anwendung oder der Client kann dann bei Bedarf Geheimnisse abrufen. Weitere Informationen finden Sie unter Was ist? AWS Secrets Manager im AWS Secrets Manager Benutzerhandbuch.

- Verwenden Sie nach Möglichkeit IAM-Rollen, um temporäre Sicherheitsanmeldeinformationen zu generieren – Verwenden Sie nach Möglichkeit immer Mechanismen, um temporäre Sicherheitsanmeldeinformationen auszugeben, anstatt langfristige Zugriffsschlüssel. Temporäre Sicherheitsanmeldeinformationen sind sicherer, da sie nicht beim Benutzer gespeichert, sondern dynamisch generiert und dem Benutzer auf Anfrage bereitgestellt werden. Temporäre Sicherheitsanmeldeinformationen haben eine begrenzte Lebensdauer, sodass Sie sie nicht verwalten oder aktualisieren müssen. Zu den Mechanismen, die temporäre Zugriffsschlüssel bereitstellen, gehören IAM-Rollen oder die Authentifizierung eines IAM-Identity-Center-Benutzers. Für Maschinen, die außerhalb von laufen, können AWS Sie AWS Identity and Access Management Roles Anywhere verwenden.
- Verwenden Sie Alternativen zu langfristigen Zugriffsschlüsseln für die AWS Command Line Interface (AWS CLI) oder die **`aws-shell`** – Zu den Alternativen gehören die folgenden.
  - AWS CloudShellist eine browserbasierte, vorab authentifizierte Shell, die Sie direkt von der aus starten können. AWS Management Console Sie können AWS CLI Befehle AWS-Services über Ihre bevorzugte Shell (Bash, Powershell oder Z-Shell) ausführen. Wenn Sie dies tun, müssen Sie keine Befehlszeilentools herunterladen oder installieren. Weitere Informationen finden Sie unter Was ist AWS CloudShell? im AWS CloudShell -Benutzerhandbuch.
  - AWS CLI Integration von Version 2 mit AWS IAM Identity Center (IAM Identity Center). Sie können Benutzer authentifizieren und kurzfristige Anmeldeinformationen für die Ausführung AWS CLI von Befehlen bereitstellen. Weitere Informationen finden Sie unter Integration AWS CLI mit IAM Identity Center im AWS IAM Identity Center Benutzerhandbuch und Konfiguration des IAM Identity Center für AWS CLI die Verwendung von IAM Identity Center im AWS Command Line Interface Benutzerhandbuch.
- Erstellen Sie keine langfristigen Zugriffsschlüssel für menschliche Benutzer, die Zugriff auf Anwendungen benötigen, oder AWS-Services — IAM Identity Center kann temporäre Zugangsdaten für den Zugriff Ihrer externen IdP-Benutzer generieren. AWS-Services Dadurch entfällt die Notwendigkeit, langfristige Anmeldeinformationen in IAM zu erstellen und zu verwalten. Erstellen Sie im IAM Identity Center eine IAM-Identity-Center-Berechtigungsgruppe, die den externen IdP-Benutzern Zugriff gewährt. Weisen Sie dann dem Berechtigungssatz in der ausgewählten Gruppe eine Gruppe aus dem IAM Identity Center zu. AWS-Konten Weitere Informationen finden Sie unter Was ist AWS IAM Identity Center, Connect zu Ihrem externen Identitätsanbieter herstellen und Berechtigungssätze im AWS IAM Identity Center Benutzerhandbuch.
- Speichern Sie langfristige Zugriffsschlüssel nicht innerhalb eines AWS Rechendienstes, sondern weisen Sie Computing-Ressourcen eine IAM-Rolle zu. Dadurch werden automatisch temporäre

Anmeldeinformationen bereitgestellt, um Zugriff zu gewähren. Wenn Sie beispielsweise ein Instance-Profil erstellen, das an eine Amazon EC2 EC2-Instance angehängt ist, können Sie der Instance eine AWS Rolle zuweisen und sie für alle ihre Anwendungen verfügbar machen. Ein Instance-Profil enthält die Rolle und ermöglicht Programmen, die auf der Amazon-EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen zu erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden.

Zugriff AWS mit Ihren Anmeldeinformationen AWS

AWS erfordert unterschiedliche Arten von Sicherheitsanmeldedaten, je nachdem, wie Sie darauf zugreifen AWS und welcher AWS Benutzertyp Sie sind. So verwenden Sie beispielsweise Anmeldeinformationen für die AWS Management Console , während Sie Zugriffsschlüssel verwenden, um programmgesteuerte Aufrufe an AWS zu tätigen. Außerdem enthält jede Identität, die Sie verwenden, unabhängig davon, ob es sich um den Root-Benutzer des Kontos, einen AWS Identity and Access Management (IAM-) AWS IAM Identity Center Benutzer, einen Benutzer oder eine föderierte Identität handelt, eindeutige Anmeldeinformationen. AWS

step-by-step Anweisungen, wie Sie sich AWS entsprechend Ihrem Benutzertyp anmelden können, finden Sie unter So melden Sie sich an AWS im AWS Anmelde-Benutzerhandbuch.

AWS Richtlinien für Sicherheitsaudits

Überprüfen Sie Ihre Sicherheitskonfiguration regelmäßig, um sich zu vergewissern, dass diese Ihren aktuellen geschäftlichen Anforderungen genügt. Eine solche Prüfung bietet Ihnen die Möglichkeit, nicht mehr benötigte IAM-Benutzer, -Rollen, -Gruppen und -Richtlinien zu entfernen und sicherzustellen, dass Ihre Benutzer und Softwareanwendungen keine übermäßigen Berechtigungen haben.

Im Folgenden finden Sie Richtlinien für die systematische Überprüfung und Überwachung Ihrer AWS Ressourcen im Hinblick auf bewährte Sicherheitsverfahren.

> **Tip**
>
> Sie können Ihre Nutzung von IAM in Bezug auf bewährte Sicherheitsmethoden überwachen, indem Sie AWS Security Hub verwenden. Security Hub verwendet Sicherheitskontrollen 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 IAM-Ressourcen finden Sie unter AWS Identity and Access Management-Steuerelemente im AWS Security Hub -Benutzerhandbuch.

Inhalt

Wann sollte eine Sicherheitsprüfung durchgeführt werden?

Überprüfen Sie Ihre Sicherheitskonfiguration in den folgenden Situatione:

- In regelmäßigen Abständen. Führen Sie als bewährte Sicherheitsmethode die in diesem Dokument beschriebenen Schritte in regelmäßigen Abständen durch.
- Wenn es in Ihrer Organisation Veränderungen gibt, z. B. Personen sie verlassen.
- Wenn Sie die Nutzung eines oder mehrerer einzelner AWS Dienste eingestellt haben, um zu überprüfen, ob Sie Berechtigungen entfernt haben, die Benutzer in Ihrem Konto nicht mehr benötigen.
- Wenn Sie Software zu Ihren Konten hinzugefügt oder entfernt haben, z. B. Anwendungen auf Amazon EC2 EC2-Instances, AWS OpsWorks Stacks, AWS CloudFormation Vorlagen usw.
- Wenn Sie vermuten, dass eine unberechtigte Person auf Ihr Konto zugegriffen hat.

Richtlinien für die Sicherheitsprüfung

Befolgen Sie bei der Prüfung der Sicherheitskonfiguration Ihres Kontos die folgenden Richtlinien:

- Seien Sie gründlich. Sehen Sie sich alle Aspekte der Sicherheitskonfiguration an, einschließlich der Funktionen, die selten verwendet verwenden.
- Stellen Sie keine Annahmen an. Wenn Sie mit bestimmten Aspekten Ihrer Sicherheitskonfiguration nicht vertraut sind (z. B. den Grund für eine bestimmte Richtlinie oder Rolle nicht kennen), gehen Sie der geschäftlichen Anforderung nach, bis Sie das potenzielle Risiko kennen.
- Halten Sie die Dinge einfach. Um die Prüfung (und Verwaltung) einfacher zu gestalten, verwenden Sie IAM-Gruppen, IAM-Rollen, konsistente Benennungen und einfache Richtlinien.

Überprüfen Sie Ihre AWS Kontoanmeldedaten

Gehen Sie wie folgt vor, wenn Sie Ihre AWS Kontoanmeldedaten überprüfen:

1. Wenn Sie die Zugriffsschlüssel für Ihren Root-Benutzer nicht verwenden, können Sie sie entfernen. Wir empfehlen dringend, für die tägliche Arbeit keine Root-Zugangsschlüssel zu verwenden AWS, sondern stattdessen Benutzer mit temporären Anmeldeinformationen zu verwenden, Benutzer im AWS IAM Identity Center z.
2. Wenn Sie Zugriffsschlüssel für Ihr Konto benötigen, stellen Sie sicher, dass Sie diese bei Bedarf aktualisieren.

Überprüfen Ihrer IAM-Benutzer

Führen Sie die folgenden Schritte aus, wenn Sie Ihre vorhandenen IAM-Benutzer prüfen:

1. Listen Sie Ihre Benutzer auf und löschen Sie dann Benutzer, die nicht notwendig sind.
2. Entfernen Sie Benutzer aus Gruppen, auf die sie keinen Zugriff benötigen.
3. Überprüfen Sie die Richtlinien, die den Gruppen zugewiesen sind, denen der Benutzer angehört. Siehe Tipps zur Prüfung der IAM-Richtlinien.
4. Löschen Sie die Sicherheitsanmeldeinformationen, die der Benutzer nicht benötigt oder die möglicherweise kompromittiert wurden. Beispielsweise benötigt ein IAM-Benutzer, der für eine Anwendung verwendet wird, kein Passwort (das nur für die Anmeldung auf AWS Websites erforderlich ist). Analog dazu gilt: Wenn ein Benutzer keine Zugriffsschlüssel verwendet, gibt es keinen Grund, dass er welche haben muss. Weitere Informationen finden Sie unter Verwalten von Passwörtern für IAM-Benutzer und Verwalten von Zugriffsschlüsseln für IAM-Benutzer.

   Sie können einen Bericht zu Anmeldeinformationen erstellen und herunterladen. In diesem Bericht sind alle IAM-Benutzer unter Ihrem Konto mit dem Status ihrer verschiedenen

Anmeldeinformationen aufgeführt (z. B. Passwörter, Zugriffsschlüssel und MFA-Geräte). Für Passwörter und Zugriffsschlüssel zeigt der Bericht zu den Anmeldeinformationen an, wann das Passwort oder der Zugriffsschlüssel zuletzt verwendet wurde. Erwägen Sie, Anmeldeinformationen, die in letzter Zeit nicht verwendet wurden, von Ihrem Konto zu entfernen. (Entfernen Sie Ihren Benutzer für den Notfallzugriff nicht.) Weitere Informationen finden Sie unter Berichte über Anmeldeinformationen für Ihr AWS Konto abrufen.

5. Aktualisieren Sie Passwörter und Zugriffsschlüssel für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern. Weitere Informationen finden Sie unter Verwalten von Passwörtern für IAM-Benutzer und Verwalten von Zugriffsschlüsseln für IAM-Benutzer.
6. Als bewährte Methode sollten menschliche Benutzer für den Zugriff AWS mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden. Wenn möglich, wechseln Sie von IAM-Benutzern zu Verbundbenutzern, z. B. Benutzern im IAM Identity Center. Behalten Sie die Mindestanzahl an IAM-Benutzern bei, die für Ihre Anwendungen erforderlich sind.

Prüfen Ihrer IAM-Gruppen

Führen Sie die folgenden Schritte aus, wenn Sie Ihre IAM-Gruppen prüfen:

1. Listen Sie Ihre Gruppen auf und löschen Sie dann Gruppen, die Sie nicht verwenden.
2. Überprüfen Sie die Benutzer in den einzelnen Gruppen und entfernen Sie die Benutzer, die dort nicht hingehören.
3. Überprüfen Sie die Richtlinien für die Gruppe. Siehe Tipps zur Prüfung der IAM-Richtlinien.

Überprüfen Ihrer IAM-Rollen

Führen Sie die folgenden Schritte aus, wenn Sie Ihre IAM-Rollen prüfen:

1. Listen Sie Ihre Rollen auf und löschen Sie dann Rollen, die Sie nicht verwenden.
2. Überprüfen Sie die Vertrauensrichtlinie der Rolle. Stellen Sie sicher, dass Sie wissen, wer der Prinzipal ist und warum das Konto bzw. der Benutzer dieser Rolle zugeordnet werden muss.
3. Überprüfen Sie die Zugriffsrichtlinie der Rolle, um sicherzugehen, dass der Person, die diese Rolle übernimmt, die richtigen Berechtigungen gewährt werden – siehe Tipps zur Prüfung der IAM-Richtlinien.

Prüfen Ihrer IAM-Anbieter für SAML oder OpenID Connect (OIDC)

Wenn Sie eine IAM-Entität zum Einrichten von Vertrauensstellungen mit einem SAML- oder OIDC-Identitätsanbieter (IDP) erstellt haben, führen Sie diese Schritte aus:

1. Löschen Sie unbenutzte Anbieter.
2. Laden Sie die AWS Metadatendokumente für jeden SAML-IdP herunter, überprüfen Sie sie und stellen Sie sicher, dass die Dokumente Ihren aktuellen Geschäftsanforderungen entsprechen.
3. Holen Sie sich die neuesten Metadatendokumente von der SAML IdPs und aktualisieren Sie den Anbieter in IAM.

Überprüfen Ihrer mobilen Apps

Wenn Sie eine mobile App erstellt haben, die Anfragen an stellt AWS, gehen Sie wie folgt vor:

1. Stellen Sie sicher, dass die mobile App keine eingebetteten Zugriffsschlüssel enthält, auch nicht in verschlüsseltem Speicher.
2. Rufen Sie die temporären Anmeldeinformationen für die App mithilfe der APIs ab, die für diesen Zweck entwickelt wurden.

> **Note**
>
> Wir empfehlen die Verwendung von Amazon Cognito zum Verwalten von Benutzeridentitäten in Ihrer App. Dieser Service ermöglicht Ihnen die Authentifizierung von Benutzern, die Login with Amazon, Facebook, Google oder einen beliebigen OpenID Connect (OIDC)-kompatiblen Identitätsanbieter nutzen. Weitere Informationen finden Sie unter Amazon-Cognito-Identitätspools im Amazon-Cognito-Entwicklerhandbuch.

Tipps zur Prüfung der IAM-Richtlinien

Die Richtlinien sind sehr wirkungsvoll und nuanciert. Deshalb ist es wichtig, die Berechtigungen genau zu kennen, die von jeder Richtlinie gewährt werden. Beachten Sie die folgenden Hinweise, wenn Sie Ihre Richtlinien überprüfen:

- Ordnen Sie Richtlinien Gruppen oder Rollen zu, anstatt einzelnen Benutzern. Wenn ein einzelner Benutzer über eine Richtlinie verfügt, müssen Sie nachvollziehen können, warum der Benutzer die Richtlinie benötigt.
- Stellen Sie sicher, dass IAM-Benutzer, -Gruppen und -Rollen über die Berechtigungen verfügen, die sie benötigen, und dass sie über keine zusätzlichen Berechtigungen verfügen.
- Verwenden Sie den IAM-Richtliniensimulator zum Testen von Richtlinien, die Benutzern oder Gruppen zugeordnet sind.
- Denken Sie daran, dass die Berechtigungen eines Benutzers das Ergebnis aller geltenden Richtlinien sind — sowohl identitätsbasierte Richtlinien (für Benutzer, Gruppen oder Rollen) als auch ressourcenbasierte Richtlinien (für Ressourcen wie Amazon S3 S3-Buckets, Amazon SQS SQS-Warteschlangen, Amazon SNS SNS-Themen und Schlüssel). AWS KMS Es ist wichtig, alle für einen Benutzer geltenden Richtlinien zu untersuchen und zu wissen, welche Berechtigungen einem einzelnen Benutzer gewährt wurden.
- Beachten Sie Folgendes: Wenn Sie einem Benutzer erlauben, IAM-Benutzer, -Gruppen, -Rollen oder -Richtlinien zu erstellen und der Prinzipalentität eine Richtlinie zuzuweisen, gewähren Sie ihm dadurch alle Berechtigungen für alle Ressourcen in Ihrem Konto. Benutzer, die Richtlinien erstellen und sie einem Benutzer, einer Gruppe oder einer Rolle zuweisen dürfen, können sich auch selbst beliebige Berechtigungen erteilen. Gewähren Sie generell keinen Benutzern oder Rollen, denen Sie nicht vertrauen, IAM-Berechtigungen mit uneingeschränktem Zugriff auf die Ressourcen unter Ihrem Konto. Vergewissern Sie sich bei der Durchführung Ihres Sicherheitsaudits, dass vertrauenswürdigen Identitäten die folgenden IAM-Berechtigungen erteilt wurden:
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Stellen Sie sicher, dass Richtlinien keine Zugriffsberechtigungen für Services erteilen, die Sie nicht nutzen. Wenn Sie beispielsweise verwaltete Richtlinien verwenden, stellen Sie sicher, dass die AWS verwalteten Richtlinien, die AWS in Ihrem Konto verwendet werden, für Dienste gelten, die Sie tatsächlich nutzen. Um herauszufinden, welche AWS verwalteten Richtlinien in Ihrem Konto

verwendet werden, verwenden Sie die `GetAccountAuthorizationDetails`IAM-API (AWS CLI Befehl: `aws iam get-account-authorization-details`).

- Wenn die Richtlinie einen Benutzer berechtigt, eine Amazon-EC2-Instance zu starten, wird möglicherweise auch die Aktion `iam:PassRole` zugelassen. Wenn dies der Fall ist, sollten explizit die Rollen aufgelistet werden, die der Benutzer an die Amazon-EC2-Instance übergeben darf.
- Überprüfen Sie alle Werte für das `Action`- oder `Resource`-Element, die `*` enthalten. Gewähren Sie nach Möglichkeit `Allow`-Zugriff auf die einzelnen Aktionen und Ressourcen, die Benutzer benötigen. Im Folgenden sind jedoch einige Gründe aufgeführt, warum es angemessen sein kann, `*` in einer Richtlinie zu verwenden:
  - Die Richtlinie erteilt administrative Berechtigungen.
  - Das Platzhalterzeichen wird der Einfachheit halber für eine Reihe ähnlicher Aktionen verwendet (z. B. `Describe*`). Sie sollten alle Aktionen kennen, die auf diese Weise referenziert werden.
  - Das Platzhalterzeichen wird verwendet, um eine Klasse von Ressourcen oder einen Ressourcenpfad anzugeben (z. B. `arn:aws:iam::`*`account-id`*`:users/division_abc/*`). Sie können Zugriff auf alle Ressourcen in dieser Klasse oder in diesem Pfad erteilen.
  - Ein Serviceaktion unterstützt keine Berechtigungen auf Ressourcenebene. Die einzige Wahl für eine Ressource ist `*`.
- Prüfen Sie die Richtliniennamen, um sicherzustellen, dass sie den Zweck der Richtlinie wiedergeben. So könnte beispielsweise eine Richtlinie einen Namen haben, der "schreibgeschützt" enthält, tatsächlich gewährt sie jedoch Schreib- oder Änderungsrechte.

Weitere Informationen zur Planung Ihres Sicherheitsaudits finden Sie unter Bewährte Methoden für Sicherheit, Identität und Compliance im AWS Architecture Center.

Datenschutz in AWS Identity and Access Management

Das AWS Modell der gilt für den Datenschutz in AWS Identity and Access Management. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der alle Systeme laufen AWS Cloud. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit IAM oder anderen AWS-Services über die Konsole, API oder SDKs arbeiten. AWS CLI AWS Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Datenverschlüsselung in IAM und AWS STS

Die Datenverschlüsselung fällt normalerweise in zwei Kategorien: Verschlüsselung im Ruhezustand und Verschlüsselung während der Übertragung.

Verschlüsselung im Ruhezustand

Daten, die von IAM erfasst und gespeichert werden, werden im Ruhezustand verschlüsselt.

- IAM – Die in gesammelten und gespeicherten Daten umfassen IP-Adressen, Metadaten des Kundenkontos und Kundenidentifizierungsdaten, einschließlich Passwörter. Metadaten für

Kundenkonten und Kundenidentifizierungsdaten werden im Ruhezustand mit AES 256 und SHA 256 für Hashes verschlüsselt.

- AWS STS— sammelt AWS STS keine Kundeninhalte, mit Ausnahme von Serviceprotokollen, in denen erfolgreiche, fehlerhafte und fehlerhafte Anfragen an den Service protokolliert werden.

Verschlüsselung während der Übertragung

Kundenidentifizierungsdaten, einschließlich Passwörter, werden während der Übertragung mit TLS 1.2 und 1.3 verschlüsselt. Alle AWS STS Endgeräte unterstützen HTTPS für die Verschlüsselung von Daten während der Übertragung. Eine Liste der AWS STS Endpunkte finden Sie unter. Regionen und Endpunkte

Schlüsselverwaltung in IAM und AWS STS

Sie können Verschlüsselungsschlüssel nicht mit IAM oder verwalten. AWS STS Weitere Informationen zu Verschlüsselungsschlüsseln finden Sie unter Was ist AWS KMS? im AWS Key Management Service Entwicklerhandbuch

Datenschutz im Netzwerkverkehr in IAM und AWS STS

Anforderungen an IAM müssen mit TLS (Transport Layer Security Protocol) gestellt werden. Sie können Verbindungen zum AWS STS Service mithilfe von VPC-Endpunkten sichern. Weitere Informationen hierzu finden Sie unter Schnittstellen-VPC-Endpunkte.

Einloggen und Überwachen AWS Identity and Access Management

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von AWS Identity and Access Management (IAM), AWS Security Token Service (AWS STS) und Ihren anderen AWS Lösungen. AWS bietet verschiedene Tools zur Überwachung Ihrer AWS Ressourcen und zur Reaktion auf potenzielle Vorfälle:

- AWS CloudTrailerfasst alle API-Aufrufe für IAM und AWS STS als Ereignisse, einschließlich Aufrufe von der Konsole und API-Aufrufe. Weitere Informationen zur Verwendung CloudTrail mit IAM und finden Sie AWS STS unter. Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail Weitere Informationen zu CloudTrail finden Sie im AWS CloudTrail Benutzerhandbuch.
- AWS Identity and Access Management Access Analyzer hilft Ihnen dabei, die Ressourcen in Ihrer Organisation und Konten, wie Amazon S3 S3-Buckets oder IAM-Rollen, zu identifizieren,

die mit einer externen Entität gemeinsam genutzt werden. Dies hilft Ihnen, unbeabsichtigten Zugriff auf Ihre Ressourcen und Daten zu identifizieren, was ein Sicherheitsrisiko darstellt. Weitere Informationen finden Sie unter Was ist IAM Access Analyzer?

- Amazon CloudWatch überwacht Ihre AWS Ressourcen und die Anwendungen, auf denen Sie laufen, AWS in Echtzeit. Sie können Kennzahlen erfassen und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme festlegen, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine bestimmte Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Sie können beispielsweise die CPU-Auslastung oder andere Kennzahlen Ihrer Amazon EC2 EC2-Instances CloudWatch verfolgen und bei Bedarf automatisch neue Instances starten. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.
- Amazon CloudWatch Logs unterstützt Sie bei der Überwachung, Speicherung und dem Zugriff auf Ihre Protokolldateien von Amazon EC2 EC2-Instances und anderen Quellen. CloudTrail CloudWatch Logs können Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.

Weitere Ressourcen und bewährte Sicherheitsmethoden für IAM finden Sie unter Bewährte Methoden und Anwendungsfälle für Sicherheit in AWS Identity and Access Management.

Compliance-Validierung für AWS Identity and Access Management

Externe Prüfer bewerten die Sicherheit und Konformität von AWS Identity and Access Management (IAM) im Rahmen mehrerer AWS Compliance-Programme. Hierzu zählen unter anderem SOC, PCI, FedRAMP, ISO und andere.

Informationen darüber, ob AWS-Service ein in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter AWS-Services 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 Implementierung 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.

Resilienz in AWS Identity and Access Management

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. AWS Regionen verfügen über mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

AWS Identity and Access Management (IAM) und AWS Security Token Service (AWS STS) sind autarke, regionale Dienste, die weltweit verfügbar sind.

IAM ist von entscheidender Bedeutung. AWS-Service Jeder in ausgeführte Vorgang AWS muss von IAM authentifiziert und autorisiert werden. IAM überprüft jede Anforderung anhand der in IAM gespeicherten Identitäten und Richtlinien, um festzustellen, ob die Anforderung erlaubt oder abgelehnt wird. IAM wurde mit einer separaten Steuerebene und Datenebene erstellt, sodass sich der Service auch bei unerwarteten Ausfällen authentifiziert. IAM-Ressourcen, die in Berechtigungen verwendet werden, wie Rollen und Richtlinien, werden in der Steuerebene gespeichert. IAM-Kunden können die Konfiguration dieser Ressourcen mithilfe von IAM-Vorgängen wie `DeletePolicy` und `AttachRolePolicy` ändern. Diese Anforderungen an Konfigurationsänderungen gehen an die Steuerebene. Es gibt eine IAM-Steuerebene für alle kommerziellen Geräte AWS-Regionen, die sich in der Region USA Ost (Nord-Virginia) befindet. Das IAM-System propagiert dann Konfigurationsänderungen auf die IAM-Datenebenen in jeder aktivierten AWS-Region. Die IAM-Datenebene ist im Wesentlichen eine schreibgeschützte Replik der Konfigurationsdaten der IAM-Steuerebene. Jede Instanz AWS-Region verfügt über eine völlig unabhängige Instanz der IAM-Datenebene, die die Authentifizierung und Autorisierung für Anfragen aus derselben Region durchführt. In jeder Region ist die IAM-Datenebene über mindestens drei Availability Zones verteilt und verfügt über ausreichende Kapazität, um den Verlust einer Availability Zone ohne Beeinträchtigung für den Kunden zu tolerieren. Sowohl die IAM-Steuer- als auch die Datenebene wurden für Null geplante Ausfallzeit entwickelt, wobei alle Software-Updates und Skalierungsvorgänge auf eine Weise ausgeführt werden, die für Kunden unsichtbar ist.

AWS STS Anfragen werden standardmäßig immer an einen einzigen globalen Endpunkt gesendet. Sie können einen regionalen AWS STS -Endpunkt verwenden, um die Latenz zu reduzieren oder

zusätzliche Redundanz für Ihre Anwendungen bereitzustellen. Weitere Informationen hierzu finden Sie unter Verwaltung AWS STS in einem AWS-Region.

Bestimmte Ereignisse können die Kommunikation zwischen Benutzern AWS-Regionen über das Netzwerk unterbrechen. Selbst wenn Sie nicht mit dem globalen IAM-Endpunkt kommunizieren können, AWS STS können Sie trotzdem IAM-Prinzipale authentifizieren und IAM kann Ihre Anfragen autorisieren. Die spezifischen Details eines Ereignisses, das die Kommunikation unterbricht, bestimmen, ob Sie auf Dienste zugreifen können. AWS In den meisten Situationen können Sie weiterhin IAM-Anmeldeinformationen in Ihrer AWS Umgebung verwenden. Die folgenden Bedingungen können für ein Ereignis gelten, das die Kommunikation unterbricht.

Zugriffsschlüsseln für IAM-Benutzer

Sie können sich auf unbestimmte Zeit in einer Region mit langfristigen Zugriffsschlüsseln für IAM-Benutzer authentifizieren. Wenn Sie die APIs AWS Command Line Interface und verwenden, können Sie AWS Zugriffsschlüssel bereitstellen, mit denen Sie Ihre Identität bei programmatischen Anfragen überprüfen AWS können.

> **Important**
>
> Als bewährte Methode empfehlen wir, dass sich Ihre Benutzer mit temporären Anmeldeinformationen, anstelle von langfristigen Zugriffsschlüsseln, anmelden.

Temporäre Anmeldeinformationen

Sie können beim AWS STS regionalen Service-Endpunkt neue temporäre Anmeldeinformationen für mindestens 24 Stunden anfordern. Die folgenden API-Operationen generieren temporäre Anmeldeinformationen.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

Prinzipale und Berechtigungen

- Möglicherweise können Sie in IAM keine Prinzipale oder Berechtigungen hinzufügen, ändern oder entfernen.

- Ihre Anmeldeinformationen spiegeln möglicherweise keine Änderungen an Ihren Berechtigungen wider, die Sie kürzlich in IAM angewendet haben. Weitere Informationen finden Sie unter Änderungen, die ich vornehme, sind nicht immer direkt sichtbar.

AWS Management Console

- Möglicherweise können Sie einen regionalen Anmelde-Endpunkt verwenden, um sich in der AWS Management Console als IAM-Benutzer anzumelden. Regionale Anmelde-Endpunkte haben das folgende URL-Format.

  `https://{Account ID}.signin.aws.amazon.com/console?region={Region}`

  Beispiel: https://111122223333.signin.aws.amazon.com/console?region=us-west-2
- Möglicherweise können Sie die Universal 2nd Factor (U2F) Multi-Faktor-Authentifizierung (MFA) nicht abschließen.

Bewährte Methoden für IAM-Resilienz

AWS hat Ausfallsicherheit in AWS-Regionen Availability Zones eingebaut. Wenn Sie die folgenden bewährten IAM-Methoden in den Systemen beachten, die mit Ihrer Umgebung interagieren, profitieren Sie von dieser Resilienz.

1. Verwenden Sie einen AWS STS regionalen Dienstendpunkt anstelle des standardmäßigen globalen Endpunkts.
2. Überprüfen Sie die Konfiguration Ihrer Umgebung auf wichtige Ressourcen, die routinemäßig IAM-Ressourcen erstellen oder ändern, und bereiten Sie eine Fallback-Lösung vor, die vorhandene IAM-Ressourcen nutzt.

Sicherheit der Infrastruktur in AWS Identity and Access Management

Als verwalteter Dienst ist AWS Identity and Access Management 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 IAM 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.

Der Zugriff auf IAM erfolgt programmatisch über die IAM HTTPS API, mit der Sie HTTPS-Anfragen direkt an den Dienst richten können. Die Abfrage-API gibt vertrauliche Informationen, einschließlich der Sicherheitsanmeldeinformationen, zurück. Daher müssen Sie bei allen API-Anforderungen HTTPS verwenden. Wenn Sie die HTTPS-API nutzen, müssen Sie Code zur digitalen Signierung von Anfragen über Ihre Anmeldeinformationen einsetzen.

Diese API-Operationen lassen sich von einem beliebigen Netzwerkstandort aus aufrufen. Da IAM jedoch ressourcenbasierte Zugriffsrichtlinien unterstützt, kann es zu Einschränkungen bezüglich der Quell-IP-Adresse kommen. Sie können auch IAM-Richtlinien verwenden, um den Zugriff über bestimmte Amazon Virtual Private Cloud (Amazon VPC)-Endpunkte oder bestimmte VPCs zu steuern. Dadurch wird der Netzwerkzugriff auf eine bestimmte IAM-Ressource effektiv nur von der spezifischen VPC innerhalb des Netzwerks isoliert. AWS

Konfiguration und Schwachstellenanalyse in AWS Identity and Access Management

AWS bewältigt grundlegende Sicherheitsaufgaben wie das Patchen von Gastbetriebssystemen (OS) und Datenbanken, die Firewallkonfiguration und die Notfallwiederherstellung. Diese Verfahren wurden von qualifizierten Dritten überprüft und zertifiziert. Weitere Informationen finden Sie in den folgenden Ressourcen:

- Modell der übergreifenden Verantwortlichkeit
- Amazon Web Services: Übersicht über Sicherheitsverfahren (Whitepaper)

Die folgenden Ressourcen befassen sich auch mit der Konfiguration und Schwachstellenanalyse in AWS Identity and Access Management (IAM):

- Compliance-Validierung für AWS Identity and Access Management
- Bewährte Methoden und Anwendungsfälle für Sicherheit in AWS Identity and Access Management

AWS verwaltete Richtlinien für AWS Identity and Access Management Access Analyzer

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet AWS wird. AWS Verwaltete Richtlinien dienen dazu, Berechtigungen für viele gängige Anwendungsfälle bereitzustellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie für alle AWS Kunden verfügbar sind. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie kundenverwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn eine neue Richtlinie eingeführt AWS-Service wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

IAM-Zugriff ReadOnly

Verwenden der `IAMReadOnlyAccess` verwalteten Richtlinie, um schreibgeschützten Zugriff auf IAM-Ressourcen zu gewähren. Diese Richtlinie gewährt die Berechtigung zum Abrufen und Auflisten aller IAM-Ressourcen. Sie ermöglicht das Anzeigen von Details und Aktivitätsberichten für Benutzer, Gruppen, Rollen, Richtlinien, Identitätsanbieter und MFA-Geräte. Sie umfasst nicht die Möglichkeit, Ressourcen zu erstellen oder zu löschen oder auf IAM Access Analyzer Ressourcen zuzugreifen.

Zeigen Sie die Richtlinie für die vollständige Liste der Datenbank-Services und Aktionen an, die von dieser Richtlinie unterstützt werden.

IAM-Passwort UserChange

Verwenden Sie diese `IAMUserChangePassword`-IAM-Richtlinie, um Benutzern das Ändern ihres eigenen Konsolen-Passworts zu ermöglichen.

Sie konfigurieren Ihre IAM-Kontoeinstellungen und die Passwortrichtlinie, damit IAM-Benutzer ihr IAM-Kontopasswort ändern können. Wenn Sie diese Aktion zulassen, fügt IAM jedem Benutzer die folgende Richtlinie an:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM AccessAnalyzer FullAccess

Verwenden Sie die `IAMAccessAnalyzerFullAccess` AWS verwaltete Richtlinie, um Ihren Administratoren den Zugriff auf IAM Access Analyzer zu ermöglichen.

Berechtigungsgruppierungen

Diese Richtlinie ist in Anweisungen gruppiert, die auf den bereitgestellten Berechtigungen basieren.

- IAM Access Analyzer – Ermöglicht vollständige Administratorberechtigungen für alle Ressourcen in IAM Access Analyzer.
- Dienstverknüpfte Rolle erstellen – Ermöglicht dem Administrator das Erstellen einesService-verknüpfteIAM-RolleIn diesem Fall kann IAM Access Analyzer Ressourcen in anderen Services in Ihrem Namen analysieren. Mit dieser Berechtigung können Sie die dienstverknüpfte Rolle nur für die Verwendung durch IAM Access Analyzer erstellen.
- AWS Organizations – Ermöglicht Administratoren die Verwendung von IAM Access Analyzer für eine Organisation in AWS Organizations. Nachdem der vertrauenswürdige Zugriff für IAM Access Analyzer aktiviert wurde AWS Organizations, können Mitglieder des Verwaltungskontos die Ergebnisse in ihrer gesamten Organisation einsehen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
```

```
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

IAM-Zugriff AccessAnalyzer ReadOnly

Verwenden Sie die `IAMAccessAnalyzerReadOnlyAccess` AWS verwaltete Richtlinie, um den schreibgeschützten Zugriff auf IAM Access Analyzer zu ermöglichen.

Um auch den schreibgeschützten Zugriff auf IAM Access Analyzer für zu ermöglichen AWS Organizations, erstellen Sie eine vom Kunden verwaltete Richtlinie, die die Aktionen Beschreiben und Auflisten aus der verwalteten Richtlinie ermöglicht. IAM AccessAnalyzer FullAccess AWS

Service Level-Berechtigungen

Diese Richtlinie gewährt schreibgeschützten Zugriff auf IAM Access Analyzer. In dieser Richtlinie sind keine anderen Dienstberechtigungen enthalten.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRoleRichtlinie

Sie können keine Verbindungen AccessAnalyzerServiceRolePolicy zu Ihren IAM-Entitäten herstellen. Diese Richtlinie ist mit einer dienstverknüpften Rolle verbunden, die es IAM Access Analyzer ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter Verwenden von dienstbezogenen Rollen für AWS Identity and Access Management Access Analyzer.

Berechtigungsgruppierungen

Diese Richtlinie ermöglicht den Zugriff auf IAM Access Analyzer, um Ressourcen-Metadaten von mehreren AWS-Services zu analysieren.

- Amazon DynamoDB — Ermöglicht Berechtigungen zum Anzeigen von DynamoDB-Streams und -Tabellen.
- Amazon Elastic Compute Cloud – Ermöglicht Berechtigungen zur Beschreibung von IP-Adressen, Snapshots und VPCs.
- Amazon Elastic Container Registry – Ermöglicht Berechtigungen zur Beschreibung von Image-Repositorys und zum Abrufen von Repository-Richtlinien.
- Amazon Elastic File System – Ermöglicht Berechtigungen zur Anzeige der Beschreibung eines Amazon-EFS-Dateisystems und die Anzeige der Richtlinie auf Ressourcenebene für ein Amazon-EFS-Dateisystem.
- AWS Identity and Access Management – Ermöglicht Berechtigungen zum Abrufen von Informationen zu einer angegebenen Rolle und zum Auflisten der IAM-Rollen mit einem angegebenen Pfad-Präfix. Ermöglicht das Abrufen von Informationen über Benutzer, Benutzergruppen, Anmeldeprofile, Zugriffsschlüssel und Daten, auf die der Service zuletzt zugegriffen hat.
- AWS Key Management Service – Ermöglicht Berechtigungen zur Anzeige von detaillierten Informationen zu einem KMS-Schlüssel und seinen wichtigsten Richtlinien und Zuschüssen.
- AWS Lambda – Ermöglicht Berechtigungen zur Anzeige von Informationen über Lambda-Aliase, Funktionen, Ebenen und Aliase.
- AWS Organizations— Erlaubt Organizations Berechtigungen und ermöglicht die Erstellung eines Analyzers innerhalb der AWS Organisation als Vertrauenszone.
- Amazon Relational Database Service – Ermöglicht Berechtigungen zur Anzeige von detaillierten Informationen zu Snapshots des Amazon RDS DB und Snapshots des Amazon-RDS-DB-Clusters.

- Amazon Simple Storage Service — Ermöglicht Berechtigungen zum Anzeigen detaillierter Informationen über Amazon S3-Zugriffspunkte, -Buckets und Amazon S3 S3-Verzeichnis-Buckets, die die Amazon S3 Express One-Speicherklasse verwenden.
- AWS Secrets Manager – Ermöglicht Berechtigungen zur Anzeige von detaillierten Informationen zu Geheimnissen und an Geheimnisse angefügten Ressourcenrichtlinien.
- Amazon Simple Notification Service – Ermöglicht Berechtigungen zur Anzeige von detaillierten Informationen zu einem Thema.
- Amazon Simple Queue Service – Ermöglicht Berechtigungen zur Anzeige von detaillierten Informationen zu bestimmten Warteschlangen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAnalyzerServiceRolePolicy",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetResourcePolicy",
        "dynamodb:ListStreams",
        "dynamodb:ListTables",
        "ec2:DescribeAddresses",
        "ec2:DescribeByoipCidrs",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcs",
        "ec2:GetSnapshotBlockPublicAccessState",
        "ecr:DescribeRepositories",
        "ecr:GetRepositoryPolicy",
        "elasticfilesystem:DescribeFileSystemPolicy",
        "elasticfilesystem:DescribeFileSystems",
        "iam:GetRole",
        "iam:ListEntitiesForPolicy",
        "iam:ListRoles",
        "iam:ListUsers",
        "iam:GetUser",
        "iam:GetGroup",
        "iam:GenerateServiceLastAccessedDetails",
        "iam:GetServiceLastAccessedDetails",
        "iam:ListAccessKeys",
```

```
        "iam:GetLoginProfile",
        "iam:GetAccessKeyLastUsed",
        "iam:ListRolePolicies",
        "iam:GetRolePolicy",
        "iam:ListAttachedRolePolicies",
        "iam:ListUserPolicies",
        "iam:GetUserPolicy",
        "iam:ListAttachedUserPolicies",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListGroupsForUser",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:ListGrants",
        "kms:ListKeyPolicies",
        "kms:ListKeys",
        "lambda:GetFunctionUrlConfig",
        "lambda:GetLayerVersionPolicy",
        "lambda:GetPolicy",
        "lambda:ListAliases",
        "lambda:ListFunctions",
        "lambda:ListLayers",
        "lambda:ListLayerVersions",
        "lambda:ListVersionsByFunction",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots",
        "rds:DescribeDBClusterSnapshotAttributes",
        "rds:DescribeDBClusterSnapshots",
        "rds:DescribeDBSnapshotAttributes",
        "rds:DescribeDBSnapshots",
        "s3:DescribeMultiRegionAccessPointOperation",
        "s3:GetAccessPoint",
        "s3:GetAccessPointPolicy",
        "s3:GetAccessPointPolicyStatus",
        "s3:GetAccountPublicAccessBlock",
```

```
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicyStatus",
        "s3:GetBucketPolicy",
        "s3:GetBucketPublicAccessBlock",
        "s3:GetMultiRegionAccessPoint",
        "s3:GetMultiRegionAccessPointPolicy",
        "s3:GetMultiRegionAccessPointPolicyStatus",
        "s3:ListAccessPoints",
        "s3:ListAllMyBuckets",
        "s3:ListMultiRegionAccessPoints",
        "s3express:GetBucketPolicy",
        "s3express:ListAllMyDirectoryBuckets",
        "sns:GetTopicAttributes",
        "sns:ListTopics",
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:ListSecrets",
        "sqs:GetQueueAttributes",
        "sqs:ListQueues"
      ],
      "Resource": "*"
    }
  ]
}
```

IAM-Access-Analyzer-Updates für verwaltete AWS -Richtlinien

Sehen Sie sich Details zu IAM-Aktualisierungen und AWS verwalteten Richtlinien an, seit der Service begonnen hat, diese Änderungen zu verfolgen. Um automatische Warnungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der IAM und Dokumentverlauf-Seite des IAM Access Analyzer.

Änderung	Beschreibung	Datum
AccessAnalyzerServiceRoleRichtlinie — Berechtigungen hinzugefügt	IAM Access Analyzer hat den Dienstebenen-Berechtigungen von Unterstüt	30. Mai 2024

Änderung	Beschreibung	Datum
	zung für Berechtigungen zum Abrufen von Informationen über IAM-Benutzer- und Rollenrichtlinien hinzugefügt. `AccessAnalyzerServiceRolePolicy`	
AccessAnalyzerServiceRolePolicy — Berechtigungen hinzugefügt	IAM Access Analyzer hat den Service-Level-Berechtigungen von Unterstützung für die Erlaubnis hinzugefügt, den aktuellen Status des Block Public Access für Amazon EC2-Snapshots abzurufen. `AccessAnalyzerServiceRolePolicy`	23. Januar 2024
AccessAnalyzerServiceRolePolicy — Berechtigungen hinzugefügt	IAM Access Analyzer hat die Service-Level-Berechtigungen von um Unterstützung für DynamoDB-Streams und -Tabellen erweitert. `AccessAnalyzerServiceRolePolicy`	11. Januar 2024
AccessAnalyzerServiceRolePolicy — Berechtigungen hinzugefügt	IAM Access Analyzer hat die Service-Level-Berechtigungen von um Unterstützung für Amazon S3 S3-Verzeichnis-Buckets erweitert. `AccessAnalyzerServiceRolePolicy`	1. Dezember 2023

Änderung	Beschreibung	Datum
IAM-Zugriff AccessAnalyzer ReadOnly — Berechtigungen hinzugefügt	Für IAM Access Analyzer gibt es neue Berechtigungen, mit denen Sie überprüfen können, ob Aktualisierungen Ihrer Richtlinien zusätzlichen Zugriff gewähren. Diese Berechtigung wird von IAM Access Analyzer benötigt, um Richtlinienprüfungen für Ihre Richtlinien durchzuführen.	26. November 2023
AccessAnalyzerServiceRoleRichtlinie — Zusätzliche Berechtigungen	Bei IAM Access Analyzer gibt es jetzt Aktionen bei den Service-Level-Berechtigungen von `AccessAnalyzerServiceRolePolicy` , die die folgenden Aktionen unterstützen: • Auflisten von Entitäten für eine Richtlinie • Generieren von Informationen zum letzten Zugriff auf Services • Auflisten der wichtigsten Informationen über den Zugriffsschlüssel	26. November 2023

Änderung	Beschreibung	Datum
AccessAnalyzerServiceRoleRichtlinie — Berechtigungen hinzugefügt	IAM Access Analyzer hat Support für die folgenden Ressourcentypen zu den Service-Level-Berechtigungen von `AccessAnalyzerServiceRolePolicy` hinzugefügt: • Amazon-EBS-Volume-Snapshots • Amazon-ECR-Repositorys • Amazon-EFS-Dateisysteme • Amazon-RDS-DB-Snapshots • Snapshots des Amazon-RDS-DB-Clusters • Amazon SNS-Themen	25. Oktober 2022
AccessAnalyzerServiceRoleRichtlinie — Berechtigungen hinzugefügt	IAM Access Analyzer hat die `lambda:GetFunctionUrlConfig` -Aktion für die Service-Level-Berechtigungen von `AccessAnalyzerServiceRolePolicy` hinzugefügt.	6. April 2022
AccessAnalyzerServiceRoleRichtlinie — Berechtigungen hinzugefügt	IAM Access Analyzer hat neue Amazon S3 Aktionen hinzugefügt, um Metadaten zu analysieren, die mit regionübergreifenden Zugriffspunkten verknüpft sind.	2. September 2021

Änderung	Beschreibung	Datum
AccessAnalyzerReadOnlyIAM-Zugriff — Berechtigungen hinzugefügt	IAM Access Analyzer hat eine neue Aktion zur Erteilung von `ValidatePolicy` -Berechtigungen hinzugefügt, damit Sie die Richtlinienprüfungen für die Validierung verwenden können. Diese Berechtigung wird von IAM Access Analyzer benötigt, um Richtlinienprüfungen für Ihre Richtlinien durchzuführen.	16. März 2021
IAM Access Analyzer hat mit der Verfolgung von Änderungen begonnen	IAM Access Analyzer hat damit begonnen, Änderungen an seinen AWS verwalteten Richtlinien nachzuverfolgen.	1. März 2021

Benutzen AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer bietet die folgenden Funktionen:

- Analysatoren für externen Zugriff von IAM Access Analyzer helfen bei der Identifizierung von Ressourcen in Ihrer Organisation und Ihren Konten, die mit einer externen Entität geteilt werden.
- Mit den Analysatoren für ungenutzten Zugriff von IAM Access Analyzer können Sie ungenutzten Zugriff identifizieren, der in Ihrer Organisation und Ihren Konten auftritt.
- IAM Access Analyzer validiert IAM-Richtlinien anhand der Richtliniengrammatik und AWS der Best Practices.
- Mithilfe der benutzerdefinierten Richtlinienprüfungen von IAM Access Analyzer können Sie überprüfen, ob IAM-Richtlinien Ihre angegebenen Sicherheitsstandards erfüllen.
- IAM Access Analyzer generiert IAM-Richtlinien auf der Grundlage der Zugriffsaktivitäten in Ihren Protokollen. AWS CloudTrail

Identifizieren von Ressourcen, die mit einer externen Entität geteilt wurden

IAM Access Analyzer hilft Ihnen, die Ressourcen in Ihrer Organisation und Konten, wie Amazon-S3-Buckets oder IAM-Rollen, die mit einer externen Entität geteilt werden, zu identifizieren. Dies hilft Ihnen, unbeabsichtigten Zugriff auf Ihre Ressourcen und Daten zu identifizieren, was ein Sicherheitsrisiko darstellt. IAM Access Analyzer identifiziert Ressourcen, die mit externen Principals gemeinsam genutzt werden, indem er die ressourcenbasierten Richtlinien in Ihrer Umgebung anhand von logischer Argumentation analysiert. AWS Für jede Instance einer Ressource, die außerhalb Ihres Kontos geteilt wird, generiert IAM Access Analyzer ein Ergebnis. Die Ergebnisse enthalten Informationen über den Zugang und den externen Prinzipal, dem dieser gewährt wurde. Sie können Ergebnisse überarbeiten, um festzustellen, ob der Zugriff beabsichtigt und sicher ist oder ob er unbeabsichtigt ist und ein Sicherheitsrisiko darstellt. Die Ergebnisse des IAM Access Analyzers helfen Ihnen nicht nur bei der Identifizierung von Ressourcen, die mit einer externen Entität geteilt werden, sondern Sie können auch eine Vorschau anzeigen, wie sich Ihre Richtlinie auf den öffentlichen und kontoübergreifenden Zugriff auf Ihre Ressource auswirkt, bevor Sie Ressourcenberechtigungen bereitstellen. Die Ergebnisse sind in einem visuellen Übersichts-Dashboard zusammengefasst. Das Dashboard verdeutlicht die Aufteilung zwischen Ergebnissen

mit öffentlichem Zugriff und kontenübergreifendem Zugriff und bietet eine Aufschlüsselung der Ergebnisse nach Ressourcentyp. Weitere Informationen zum Dashboard finden Sie unter Anzeigen des Dashboards mit den Ergebnissen von IAM Access Analyzer.

> **Note**
>
> Eine externe Entität kann ein anderes AWS Konto, ein Root-Benutzer, ein IAM-Benutzer oder eine IAM-Rolle, ein Verbundbenutzer, ein AWS Dienst, ein anonymer Benutzer oder eine andere Entität sein, mit der Sie einen Filter erstellen können. Weitere Informationen finden Sie unter AWS -JSON-Richtlinienelemente: Auftraggeber.

Wenn Sie IAM Access Analyzer aktivieren, erstellen Sie einen Analyzer für Ihre gesamte Organisation oder Ihr Konto. Die von Ihnen gewählte Organisation oder das von Ihnen ausgewählte Konto wird als Vertrauenszone für den Analysator bezeichnet. Der Analysator überwacht alle unterstützten Ressourcen in Ihrer Vertrauenszone. Jeder Ressourcenzugang von Auftraggeber innerhalb Ihrer Vertrauenszone gilt als vertrauenswürdig. Nach der Aktivierung analysiert IAM Access Analyzer die Richtlinien, die auf alle unterstützten Ressourcen in Ihrer Vertrauenszone angewendet werden. Nach der ersten Analyse analysiert IAM Access Analyzer diese Richtlinien regelmäßig. Wenn Sie eine neue Richtlinie hinzufügen oder eine vorhandene Richtlinie ändern, analysiert IAM Access Analyzer die neue oder aktualisierte Richtlinie innerhalb von etwa 30 Minuten.

Wenn IAM Access Analyzer bei der Analyse der Richtlinien eine identifiziert, die Zugriff auf einen externen Auftraggeber gewährt, der sich nicht in Ihrer Vertrauenszone befindet, generiert es ein Ergebnis. Jedes Ergebnis enthält Details über die Ressource, die externe Entität, die Zugriff darauf hat, und die gewährten Berechtigungen, so dass Sie entsprechende Maßnahmen ergreifen können. Sie können die im Ergebnis enthaltenen Details anzeigen, um festzustellen, ob der Ressourcenzugriff beabsichtigt ist oder ein potenzielles Risiko darstellt, das Sie lösen sollten. Wenn Sie einer Ressource eine Richtlinie hinzufügen oder eine vorhandene Richtlinie aktualisieren, analysiert IAM Access Analyzer die Richtlinie. IAM Access Analyzer analysiert außerdem regelmäßig alle ressourcenbasierten Richtlinien.

In seltenen Fällen erhält IAM Access Analyzer unter bestimmten Bedingungen keine Benachrichtigung über eine hinzugefügte oder aktualisierte Richtlinie, was zu Verzögerungen bei den generierten Ergebnissen führen kann. Es kann bis zu 6 Stunden dauern, bis IAM Access Analyzer Ergebnisse generiert oder behoben hat, wenn Sie einen Access Point mit mehreren Regionen erstellen oder löschen, der einem Amazon S3 S3-Bucket zugeordnet ist, oder wenn Sie die Richtlinie für den Access Point mit mehreren Regionen aktualisieren. Wenn bei der AWS CloudTrail

Protokollzustellung ein Zustellungsproblem auftritt, löst die Änderung der Richtlinie außerdem keinen erneuten Scan der im Ergebnis gemeldeten Ressource aus. In diesem Fall analysiert IAM Access Analyzer die neue oder aktualisierte Richtlinie bei der nächsten periodischen Überprüfung, die innerhalb von 24 Stunden stattfindet. Wenn Sie bestätigen möchten, dass eine Änderung, die Sie an einer Richtlinie vorgenommen haben, ein in ein Ergebnis gemeldetes Zugriffsproblem behebt, können Sie die in ein Ergebnis gemeldete Ressource erneut scannen, indem Sie den Link Rescan (Erneut scannen) auf der Seite Findings (Ergebnisdetails) verwenden oder die `StartResourceScan`-Funktion der IAM Access Analyzer API verwenden. Weitere Informationen hierzu finden Sie unter Lösen von Ergebnissen.

> **Important**
>
> IAM Access Analyzer analysiert nur Richtlinien, die auf Ressourcen in derselben AWS Region angewendet werden, in der sie aktiviert sind. Um alle Ressourcen in Ihrer AWS Umgebung zu überwachen, müssen Sie einen Analyzer erstellen, um IAM Access Analyzer in jeder Region zu aktivieren, in der Sie unterstützte AWS Ressourcen verwenden.

IAM Access Analyzer analysiert die folgenden Ressourcentypen:

- Amazon-Simple-Storage-Service-Buckets
- Amazon-Simple-Storage-Service-Verzeichnis-Buckets
- AWS Identity and Access Management Rollen
- AWS Key Management Service Schlüssel
- AWS Lambda Funktionen und Schichten
- Amazon Simple Queue Service Warteschlangen als Ziele
- AWS Secrets Manager Geheimnisse
- Amazon Simple Notification Service-Themen
- Volume-Snapshots von Amazon Elastic Block Store
- DB-Snapshots von Amazon Relational Database Service
- Snapshots von Service-DB-Clustern von Amazon Relational Database
- Repositories der Amazon Elastic Container Registry
- Dateisysteme des Amazon Elastic File System
- Amazon DynamoDB DynamoDB-Streams

- Amazon-DynamoDB-Tabellen

Identifizieren von IAM-Benutzern und -Rollen mit gewährtem ungenutztem Zugriff

IAM Access Analyzer hilft Ihnen dabei, ungenutzten Zugriff in Ihrer AWS Organisation und Ihren Konten zu identifizieren und zu überprüfen. IAM Access Analyzer überwacht kontinuierlich alle IAM-Rollen und -Benutzer in Ihrer AWS -Organisation und Ihren -Konten und generiert Ergebnisse für ungenutzten Zugriff. Die Ergebnisse heben ungenutzte Rollen, ungenutzte Zugriffsschlüssel für IAM-Benutzer und ungenutzte Passwörter für IAM-Benutzer hervor. Für aktive IAM-Rollen und -Benutzer bieten die Ergebnisse Aufschluss über ungenutzte Services und Aktionen.

Die Ergebnisse für externen und ungenutzten Zugriff werden in einem visuellen Übersichts-Dashboard dargestellt. Das Dashboard hebt diejenigen hervor AWS-Konten , die die meisten Ergebnisse aufweisen, und bietet eine Aufschlüsselung der Ergebnisse nach Typ. Weitere Informationen zu den Seiten im Dashboard finden Sie unter Anzeigen des Dashboards mit den Ergebnissen von IAM Access Analyzer.

IAM Access Analyzer überprüft die Informationen, auf die zuletzt zugegriffen wurde, für alle Rollen in Ihrer AWS Organisation und Konten, um Ihnen zu helfen, ungenutzten Zugriff zu identifizieren. Die Informationen über den letzten Zugriff auf die IAM-Aktion helfen Ihnen dabei, ungenutzte Aktionen für Rollen in Ihrem AWS-Konten zu identifizieren. Weitere Informationen finden Sie unter Verfeinerung der Berechtigungen bei der AWS Verwendung von Informationen, auf die zuletzt zugegriffen wurde.

Überprüfung von Richtlinien anhand bewährter Methoden für AWS

Sie können Ihre Richtlinien anhand der IAM-Richtliniengrammatik und der Bewährten Methoden für AWS überprüfen, indem Sie die grundlegenden Richtlinienprüfungen der Richtlinienvalidierung von IAM Access Analyzer nutzen. Sie können eine Richtlinie mithilfe des AWS CLI AWS API- oder JSON-Richtlinieneditors in der IAM-Konsole erstellen oder bearbeiten. Sie können die Ergebnisse der Richtlinienvalidierung anzeigen, die Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge für Ihre Richtlinie enthalten. Diese Ergebnisse bieten umsetzbare Empfehlungen, die Ihnen helfen, Richtlinien zu erstellen, die funktionsfähig sind und den Best Practices entsprechen AWS . Weitere Informationen zum Validieren von Richtlinien mit der Richtlinienvalidierung finden Sie unter Richtlinienvalidierung von IAM Access Analyzer.

Überprüfen von Richtlinien anhand der von Ihnen angegebenen Sicherheitsstandards

Sie können Ihre Richtlinien mithilfe von benutzerdefinierten Richtlinienüberprüfungen von IAM Access Analyzer mit Ihren vorhandenen Sicherheitsstandards abgleichen. Sie können eine Richtlinie mithilfe des AWS API- oder JSON-Richtlinieneditors in der AWS CLI IAM-Konsole erstellen oder bearbeiten. Sie können über die Konsole überprüfen, ob die aktualisierte Richtlinie im Gegensatz zur vorhandenen Version nun Zugriff gewährt. Über AWS CLI eine AWS API können Sie auch überprüfen, ob bestimmte IAM-Aktionen, die Sie für wichtig halten, in einer Richtlinie nicht zulässig sind. Diese Überprüfungen heben eine Richtlinienanweisung hervor, die neuen Zugriff gewährt. Sie können die Richtlinienanweisung aktualisieren und die Prüfungen erneut ausführen, bis die Richtlinie Ihrem Sicherheitsstandard entspricht. Weitere Informationen zum Validieren von Richtlinien mit benutzerdefinierter Richtlinienprüfungen finden Sie unter IAM Access Analyzer – Benutzerdefinierte Richtlinienüberprüfungen.

Generieren von Richtlinien

IAM Access Analyzer analysiert Ihre AWS CloudTrail Protokolle, um Aktionen und Dienste zu identifizieren, die von einer IAM-Entität (Benutzer oder Rolle) innerhalb des von Ihnen angegebenen Datumsbereichs verwendet wurden. Es generiert dann eine IAM-Richtlinie, die auf dieser Aktivität basiert. Sie können die generierte Richtlinie verwenden, um die Berechtigungen einer Entität zu verfeinern, indem Sie sie einem IAM-Benutzer oder einer IAM-Rolle zuordnen. Weitere Informationen zum Generieren von Richtlinien mit IAM Access Analyzer finden Sie unter Generieren von IAM Access Analyzer-Richtlinien.

Preise für IAM Access Analyzer

IAM Access Analyzer erhebt Gebühren für die Analyse des ungenutzten Zugriffs. Sie entsprechen der Anzahl der pro Analysator und pro Monat analysierten IAM-Rollen und -Benutzer.

- Jeder Analysator für ungenutzten Zugriff, den Sie erstellen, wird Ihnen in Rechnung gestellt.
- Wenn Sie Analysatoren für ungenutzten Zugriff in mehreren Regionen einrichten, wird Ihnen jeder Analysator in Rechnung gestellt.
- Serviceverknüpfte Rollen werden nicht auf nicht genutzte Zugriffsaktivitäten analysiert und sind nicht in der Gesamtzahl der analysierten IAM-Rollen enthalten.

IAM Access Analyzer berechnet Gebühren für benutzerdefinierte Richtlinienprüfungen. Die Gebühren beruhen darauf, wie viele API-Anfragen an IAM Access Analyzer auf der Suche nach neuen Zugriffen gestellt werden.

Eine vollständige Liste der Kosten und Preise für IAM Access Analyzer finden Sie unter Preise für IAM Access Analyzer.

Um Ihre Rechnung anzuzeigen, navigieren Sie zu Fakturierungs- und Kostenverwaltungs-Dashboard in der AWS Billing and Cost Management -Konsole. Ihre Abrechnung enthält Links zu Nutzungsberichten mit Details zu Ihrer Abrechnung. Weitere Informationen zur AWS-Konto Abrechnung finden Sie im Benutzerhandbuch AWS Billing

Wenn Sie Fragen zu AWS Abrechnung, Konten und Veranstaltungen haben, wenden Sie sich an AWS Support.

Ergebnisse für externen und ungenutzten Zugriff

IAM Access Analyzer generiert Ergebnisse für externen Zugriff und ungenutzten Zugriff in Ihrem AWS-Konto oder Ihrer Organisation. IAM Access Analyzer generiert für externen Zugriff ein Ergebnis für jede Instance einer ressourcenbasierten Richtlinie, die einem Prinzipal, der sich nicht innerhalb Ihrer Vertrauenszone befindet, Zugriff auf eine Ressource in Ihrer Vertrauenszone gewährt. Wenn Sie einen externen Access Analyzer erstellen, wählen Sie eine Organisation oder eine Organisation aus, AWS-Konto die analysiert werden soll. Jeder Auftraggeber in der Organisation oder im Konto, den Sie für als Analysator auswählen, gilt als vertrauenswürdig. Da Auftraggeber in derselben Organisation oder demselben Konto vertrauenswürdig sind, bilden die Ressourcen und Auftraggeber innerhalb der Organisation oder des Kontos die Vertrauenszone für den Analysator. Jede Freigabe innerhalb der Vertrauenszone wird als sicher angesehen, so dass IAM Access Analyzer kein Ergebnis generiert. Wenn Sie beispielsweise eine Organisation als Vertrauenszone für einen Analysator auswählen, befinden sich alle Ressourcen und Auftraggeber in der Organisation innerhalb der Vertrauenszone. Wenn Sie einem Prinzipal in einem anderen Mitgliedskonto Ihrer Organisation Berechtigungen für einen Amazon-S3-Bucket in einem Ihrer Mitgliedskonten Ihrer Organisation erteilen, generiert IAM Access Analyzer kein Ergebnis. Wenn Sie jedoch einem Auftraggeber Zugriff auf ein Konto gewähren, das nicht Mitglied der Organisation ist, generiert IAM Access Analyzer ein Ergebnis.

IAM Access Analyzer generiert auch Ergebnisse für ungenutzten Zugriff, der in Ihrer AWS Organisation und Ihren Konten gewährt wurde. Wenn Sie einen Analyzer für ungenutzten Zugriff erstellen, überwacht IAM Access Analyzer kontinuierlich alle IAM-Rollen und -Benutzer in Ihrer AWS Organisation und Ihren Konten und generiert Ergebnisse für ungenutzten Zugriff. IAM Access Analyzer generiert die folgenden Arten von Ergebnissen für ungenutzten Zugriff:

- Ungenutzte Rollen – Rollen ohne Zugriffsaktivität innerhalb des angegebenen Nutzungsfensters.
- Ungenutzte IAM-Benutzerzugriffsschlüssel und -Passwörter – Anmeldeinformationen von IAM-Benutzern, mit denen sie auf Ihr AWS-Konto zugreifen können.
- Ungenutzte Berechtigungen – Berechtigungen auf Service- und Aktionsebene, die von einer Rolle innerhalb des angegebenen Nutzungsfensters nicht verwendet wurden. IAM Access Analyzer verwendet identitätsbasierte Richtlinien, die Rollen zugeordnet sind, um zu bestimmen, auf welche Services und Aktionen diese Rollen zugreifen können. IAM Access Analyzer unterstützt die Überprüfung ungenutzter Berechtigungen für alle Berechtigungen auf Serviceebene. Eine vollständige Liste der Berechtigungen auf Aktionsebene, die für ungenutzte Zugriffsergebnisse unterstützt werden, finden Sie unter IAM-Aktion, zuletzt aufgerufene Informationsservices und Aktionen.

> **Note**
>
> IAM Access Analyzer bietet kostenlose Ergebnisse für externen Zugriff und erhebt Gebühren für die Ergebnisse zum ungenutzten Zugriff. Sie entsprechen der Anzahl der pro Analysator und pro Monat analysierten IAM-Rollen und -Benutzer. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

Themen

- Integrieren Sie Access Analyzer mit AWS Security Hub
- Protokollieren von IAM Access Analyzer-API-Aufrufen mit AWS CloudTrail
- Filterschlüssel für IAM Access Analyzer
- Verwenden von serviceverknüpften Rollen für AWS Identity and Access Management Access Analyzer

Funktionsweise der Ergebnisse von IAM Access Analyzer

In diesem Thema werden die in IAM Access Analyzer verwendeten Konzepte und Begriffe beschrieben, damit Sie sich damit vertraut machen können, wie IAM Access Analyzer den Zugriff auf Ihre AWS Ressourcen überwacht.

Externer Zugriff

For External Access Analyzers AWS Identity and Access Management Access Analyzer basiert auf Zelkova, das IAM-Richtlinien in entsprechende logische Anweisungen übersetzt und eine Reihe von allgemeinen und speziellen logischen Solvern (Erfüllbarkeitsmodulo-Theorien) zur Lösung des Problems einsetzt. IAM Access Analyzer wendet Zelkova wiederholt auf eine Richtlinie mit immer spezifischeren Abfragen an, um Verhaltensklassen zu charakterisieren, die die Richtlinie basierend auf dem Inhalt der Richtlinie zulässt. Weitere Informationen zu den Satisfiability Modulo-Theorien finden Sie unter Satisfiability Modulo-Theorien.

Für Analysatoren für externen Zugriff überprüft IAM Access Analyzer keine Zugriffsprotokolle, um festzustellen, ob eine externe Entität auf eine Ressource innerhalb Ihrer Vertrauenszone zugegriffen hat. Es erzeugt einen Fund, wenn eine ressourcenbasierte Richtlinie den Zugriff auf eine Ressource erlaubt, auch wenn die externe Entität nicht auf die Ressource zugegriffen hat. IAM Access Analyzer berücksichtigt bei seiner Ermittlung auch nicht den Status von externen Konten. Wenn also angegeben wird, dass Konto 111122223333 auf Ihren Amazon-S3-Bucket zugreifen kann, ist nichts über den Status von Benutzern, Rollen, Service-Kontrollrichtlinien (SCP) und anderen relevanten Konfigurationen in diesem Konto bekannt. Dies dient dem Datenschutz der Kunden – IAM Access Analyzer berücksichtigt nicht, wem das andere Konto gehört. Dies dient auch der Sicherheit – wenn das Konto nicht dem Kunden von IAM Access Analyzer gehört, ist es dennoch wichtig zu wissen, dass eine externe Entität Zugriff auf die Ressourcen erhalten könnte, auch wenn derzeit keine Auftraggeber im Konto vorhanden sind, die auf die Ressourcen zugreifen könnten.

IAM Access Analyzer berücksichtigt nur bestimmte IAM-Bedingungsschlüssel, auf die externe Benutzer keinen direkten Einfluss haben oder die sich auf andere Weise auf die Autorisierung

auswirken. Beispiele für Bedingungsschlüssel, die IAM Access Analyzer berücksichtigt, finden Sie unter Filterschlüssel für IAM Access Analyzer.

IAM Access Analyzer meldet derzeit keine Ergebnisse von Service Principals oder internen Service Accounts. AWS In seltenen Fällen, in denen der IAM Access Analyzer nicht vollständig bestimmen kann, ob eine Richtlinie Zugriff auf eine externe Entität gewährt, wird der Befund fälschlicherweise als falsch positiv deklariert. IAM Access Analyzer wurde entwickelt, um einen umfassenden Überblick über die gemeinsame Nutzung von Ressourcen in Ihrem Konto zu geben, und ist bestrebt, falsch negative Ergebnisse zu minimieren.

Ungenutzter Zugriff

Sie müssen einen Analysator für Ergebnisse zu ungenutztem Zugriff für Ihre Rollen erstellen, auch wenn Sie bereits einen Analysator erstellt haben, der Ergebnisse zum externen Zugriff für Ihre Ressourcen zu generiert. Nach der Erstellung des Analysators überprüft IAM Access Analyzer die Zugriffsaktivitäten, um ungenutzten Zugriff zu erkennen. IAM Access Analyzer überprüft die zuletzt aufgerufenen Informationen für alle Rollen, Benutzerzugriffsschlüssel und Benutzerkennwörter in Ihrer AWS Organisation und Ihren Konten, um Ihnen zu helfen, ungenutzten Zugriff zu identifizieren. Für aktive IAM-Rollen und -Benutzer verwendet IAM Access Analyzer Informationen zum IAM-Service und zur Aktion, auf die zuletzt zugegriffen wurde, um ungenutzte Berechtigungen zu erkennen. Sie können Analyzer für ungenutzten Zugriff verwenden, um Ihren Überprüfungsprozess auf AWS Organisations- und Kontoebene zu skalieren. Sie können mithilfe von Informationen zur Aktion, auf die zuletzt zugegriffen wurde, einzelne Rollen genauer untersuchen.

Übersichts-Dashboard

Sowohl für externen als auch für ungenutzten Zugriff organisiert IAM Access Analyzer die Ergebnisse in einem Übersichts-Dashboard. Das Übersichts-Dashboard verdeutlicht im Fall von externem Zugriff die Aufteilung zwischen Ergebnissen mit öffentlichem Zugriff und kontoübergreifendem Zugriff und bietet eine Aufschlüsselung der Ergebnisse nach Ressourcentyp. Bei ungenutztem Zugriff hebt das Dashboard diejenigen hervor, AWS-Konten die die meisten Ergebnisse aufweisen, und bietet eine Aufschlüsselung der Ergebnisse nach Typ. Nachdem Sie einen Analysator für externen oder ungenutzten Zugriff erstellt haben, fügt IAM Access Analysator dem Dashboard automatisch neue Ergebnisse hinzu, die sich auf Rollen mit ungenutzten Berechtigungen konzentrieren.

Erste Schritte mit AWS Identity and Access Management Access Analyzer - Ergebnisse

Verwenden Sie die Informationen in diesem Thema, um mehr über die Anforderungen zu erfahren, die für die Verwendung und Verwaltung von IAM Access Analyzer erforderlich sind AWS Identity and Access Management Access Analyzer, und erfahren Sie dann, wie Sie IAM Access Analyzer aktivieren. Weitere Informationen zur serviceverknüpften Rolle für IAM Access Analyzer finden Sie unter Verwenden von serviceverknüpften Rollen für AWS Identity and Access Management Access Analyzer.

Erforderliche Berechtigungen zur Verwendung von IAM Access Analyzer

Um IAM Access Analyzer erfolgreich zu konfigurieren und zu verwenden, müssen dem von Ihnen verwendeten Konto die erforderlichen Berechtigungen erteilt werden.

AWS verwaltete Richtlinien für IAM Access Analyzer

AWS Identity and Access Management Access Analyzer bietet AWS verwaltete Richtlinien, damit Sie schnell loslegen können.

- IAM AccessAnalyzer FullAccess — Ermöglicht Administratoren vollen Zugriff auf IAM Access Analyzer. Diese Richtlinie ermöglicht auch die Erstellung der dienstbezogenen Rollen, die erforderlich sind, damit IAM Access Analyzer Ressourcen in Ihrem Konto oder Ihrer Organisation analysieren kann. AWS
- AccessAnalyzerReadOnlyIAM-Zugriff — Ermöglicht den schreibgeschützten Zugriff auf IAM Access Analyzer. Sie müssen zusätzliche Richtlinien zu Ihren IAM-Identitäten (-Benutzer, -Benutzergruppen oder -Rollen) hinzufügen, damit diese die ihre Ergebnisse anzeigen können.

Von IAM Access Analyzer definierte Ressourcen

Informationen zu den von Access Analyzer definierten Ressourcen finden Sie unter Von IAM Access Analyzer definierte Ressourcentypen in der Service-Autorisierungsreferenz.

Erforderliche Service-Berechtigungen für IAM Access Analyzer

IAM Access Analyzer verwendet eine servicegebundene IAM-Rolle (SLR) namens `AWSServiceRoleForAccessAnalyzer`. Diese Spiegelreflexkamera gewährt dem Dienst nur Lesezugriff, um AWS Ressourcen mit ressourcenbasierten Richtlinien zu analysieren und

ungenutzten Zugriff in Ihrem Namen zu analysieren. In den folgenden Szenarien erstellt der Service die Rolle in Ihrem Konto:

- Sie erstellen einen Analysator für externen Zugriff mit Ihrem Konto als Vertrauenszone.
- Sie erstellen einen Analysator für ungenutzten Zugriff mit Ihrem Konto als ausgewähltem Konto.

Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für AWS Identity and Access Management Access Analyzer.

> **Note**
>
> IAM Access Analyzer ist regional. Für externen Zugriff müssen Sie IAM Access Analyzer in jeder Region separat aktivieren.
> Bei ungenutztem Zugriff ändern sich die Ergebnisse für den Analysator nicht von Region zu Region. Es ist nicht erforderlich, in jeder Region, in der Sie über Ressourcen verfügen, einen Analysator zu erstellen.

In einigen Fällen wird nach der Erstellung eines Analysators für externen Zugriff oder ungenutzten Zugriff in IAM Access Analyzer die Seite Ergebnisse oder das Dashboard ohne Ergebnisse oder Zusammenfassung geladen. Dies kann auf eine Verzögerung in der Konsole beim Auffüllen Ihrer Ergebnisse zurückzuführen sein. Möglicherweise müssen Sie den Browser manuell aktualisieren oder später erneut nachschauen, um die Ergebnisse oder die Zusammenfassung anzuzeigen. Wenn Sie immer noch keine Ergebnisse für einen Analysator für externen Zugriff sehen, liegt das daran, dass die unterstützten Ressourcen in Ihrem Konto fehlen, auf die eine externe Entität zugreifen kann. Wenn eine Richtlinie, die einer externen Entität Zugriff gewährt, auf eine Ressource angewendet wird, generiert IAM Access Analyzer ein Ergebnis.

> **Note**
>
> Bei Analysatoren für externen Zugriff kann es nach Änderung einer Richtlinie bis zu 30 Minuten dauern, bis IAM Access Analyzer die Ressource analysiert und dann entweder ein neues Ergebnis mit externem Zugriff generiert oder ein vorhandenes Ergebnis für den Zugriff auf die Ressource aktualisiert. Bei Analysatoren für externen und ungenutzten Zugriff erscheinen die aktuellen Ergebnisse möglicherweise nicht sofort im Dashboard.

Erforderliche Berechtigungen für IAM Access Analyzer zum Anzeigen des Ergebnis-Dashboards

Um das Ergebnis-Dashboard von IAM Access Analyzer anzuzeigen, braucht das von Ihnen verwendete Konto Zugriff für die folgenden erforderlichen Aktionen:

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Informationen zu alle von Access Analyzer definierten Aktionen finden Sie unter Von IAM Access Analyzer definierte Aktionstypen in der Service-Autorisierungsreferenz.

Aktivieren von IAM Access Analyzer

Um einen externen Zugriffsanalysator mit der AWS-Konto Vertrauenszone zu erstellen

Um einen Analysator für externen Zugriff in einer Region zu aktivieren, müssen Sie einen Analysator in dieser Region erstellen. Sie müssen in jeder Region, in der Sie den Zugriff auf Ihre Ressourcen überwachen möchten, einen Analysator für externen Zugriff erstellen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Auf Analysator zugreifen.
3. Wählen Sie Analysator-Einstellungen.
4. Wählen Sie Analysator erstellen.
5. Wählen Sie im Abschnitt Analyse die Option Analyse des externen Zugriffs aus.
6. Kontrollieren Sie im Abschnitt Analyzer-Details, ob es sich bei der angezeigten Region um die Region handelt, in der Sie IAM Access Analyzer aktivieren möchten.
7. Geben Sie einen Namen für den Analysator ein.
8. Wählen Sie Aktuelles AWS-Konto als Vertrauenszone für den Analysator.

   > **Note**
   >
   > Wenn es sich bei Ihrem Konto nicht um das AWS Organizations Verwaltungskonto oder das delegierte Administratorkonto handelt, können Sie nur einen Analyzer mit Ihrem Konto als Vertrauenszone erstellen.

9. Optional. Fügen Sie alle Tags hinzu, die auf den Analysator angewendet werden sollen.

10. Wählen Sie Absenden aus.

Wenn Sie einen Analysator für externen Zugriff erstellen, um IAM Access Analyzer zu aktivieren, wird in Ihrem Konto eine serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAccessAnalyzer` erstellt.

So erstellen Sie einen Analysator für externen Zugriff mit der Organisation als Vertrauenszone

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Auf Analysator zugreifen.
3. Wählen Sie Analysator-Einstellungen.
4. Wählen Sie Analysator erstellen.
5. Wählen Sie im Abschnitt Analyse die Option Analyse des externen Zugriffs aus.
6. Kontrollieren Sie im Abschnitt Analyzer-Details, ob es sich bei der angezeigten Region um die Region handelt, in der Sie IAM Access Analyzer aktivieren möchten.
7. Geben Sie einen Namen für den Analysator ein.
8. Wählen Sie Aktuelle Organisation als Vertrauenszone für den Analysator.
9. Optional. Fügen Sie alle Tags hinzu, die auf den Analysator angewendet werden sollen.
10. Wählen Sie Absenden aus.

Wenn Sie einen Analysator für externen Zugriff mit der Organisation als Vertrauenszone erstellen, wird in jedem Konto Ihrer Organisation eine servicegebundene Rolle mit dem Namen `AWSServiceRoleForAccessAnalyzer` erstellt.

So erstellen Sie einen Analysator für ungenutzten Zugriff für das aktuelle Konto

Gehen Sie wie folgt vor, um einen Analysator für ungenutzten Zugriff für ein einzelnes AWS-Konto zu erstellen. Bei ungenutztem Zugriff ändern sich die Ergebnisse für den Analysator nicht von Region zu Region. Es ist nicht erforderlich, in jeder Region, in der Sie über Ressourcen verfügen, einen Analysator zu erstellen.

IAM Access Analyzer erhebt Gebühren für die Analyse des ungenutzten Zugriffs. Sie entsprechen der Anzahl der pro Monat und pro Analysator analysierten IAM-Rollen und -Benutzer. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

2. Wählen Sie Auf Analysator zugreifen.
3. Wählen Sie Analysator-Einstellungen.
4. Wählen Sie Analysator erstellen.
5. Wählen Sie im Abschnitt Analyse die Option Ungenutzte Zugriffsanalyse aus.
6. Geben Sie einen Namen für den Analysator ein.
7. Geben Sie unter Nachverfolgungszeitraum die Anzahl der Tage ein, für die Ergebnisse für ungenutzte Berechtigungen generiert werden sollen. Wenn Sie beispielsweise 90 Tage eingeben, generiert der Analysator Ergebnisse für IAM-Entitäten innerhalb des ausgewählten Kontos und berücksichtigt dabei alle Berechtigungen, die innerhalb von 90 oder mehr Tagen seit dem letzten Scan des Analysators nicht verwendet wurden. Sie können einen Wert zwischen 1 und 180 Tagen wählen.
8. Wählen Sie für Ausgewählte Konten die Option Aktuelles AWS-Konto aus.

> **Note**
>
> Wenn es sich bei Ihrem Konto nicht um das AWS Organizations Verwaltungskonto oder das delegierte Administratorkonto handelt, können Sie nur einen Analyzer mit Ihrem Konto als ausgewähltem Konto erstellen.

9. Optional. Fügen Sie alle Tags hinzu, die auf den Analysator angewendet werden sollen.
10. Wählen Sie Absenden aus.

Wenn Sie einen Analysator für ungenutzten Zugriff erstellen, um IAM Access Analyzer zu aktivieren, wird in Ihrem Konto eine serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAccessAnalyzer` erstellt.

So erstellen Sie einen Analysator für ungenutzten Zugriff mit der aktuellen Organisation

Gehen Sie wie folgt vor, um einen Analysator für ungenutzten Zugriff für eine Organisation zu erstellen, mit dem Sie alle Daten AWS-Konten in einer Organisation zentral überprüfen können. Bei der Analyse für ungenutzten Zugriff ändern sich die Ergebnisse für den Analysator nicht von Region zu Region. Es ist nicht erforderlich, in jeder Region, in der Sie über Ressourcen verfügen, einen Analysator zu erstellen.

IAM Access Analyzer erhebt Gebühren für die Analyse des ungenutzten Zugriffs. Sie entsprechen der Anzahl der pro Monat und pro Analysator analysierten IAM-Rollen und -Benutzer. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

> **Note**
>
> Wenn ein Mitgliedskonto aus der Organisation entfernt wird, generiert der Analysator für ungenutzten Zugriff nach 24 Stunden keine neuen Ergebnisse mehr und aktualisiert keine vorhandenen Ergebnisse mehr für dieses Konto. Ergebnisse im Zusammenhang mit dem Mitgliedskonto, das aus der Organisation entfernt wurde, werden nach 90 Tagen dauerhaft entfernt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Auf Analysator zugreifen.
3. Wählen Sie Analysator-Einstellungen.
4. Wählen Sie Analysator erstellen.
5. Wählen Sie im Abschnitt Analyse die Option Ungenutzte Zugriffsanalyse aus.
6. Geben Sie einen Namen für den Analysator ein.
7. Geben Sie unter Nachverfolgungszeitraum die Anzahl der Tage ein, für die Ergebnisse für ungenutzte Berechtigungen generiert werden sollen. Wenn Sie beispielsweise 90 Tage eingeben, generiert der Analysator Ergebnisse für IAM-Entitäten innerhalb der Konten der ausgewählten Organisation und berücksichtigt dabei alle Berechtigungen, die innerhalb von 90 oder mehr Tagen seit dem letzten Scan des Analysators nicht verwendet wurden. Sie können einen Wert zwischen 1 und 180 Tagen wählen.
8. Wählen Sie für Ausgewählte Konten die Option Aktuelle Organisation als ausgewählte Konten für den Analysator aus.
9. Optional. Fügen Sie alle Tags hinzu, die auf den Analysator angewendet werden sollen.
10. Wählen Sie Absenden aus.

Wenn Sie einen Analysator für ungenutzten Zugriff erstellen, um IAM Access Analyzer zu aktivieren, wird in Ihrem Konto eine serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAccessAnalyzer` erstellt.

Status von IAM Access Analyzer

Um den Status Ihrer Analysatoren anzuzeigen, wählen Sie Analysatoren. Analysatoren, die für eine Organisation oder ein Konto erstellt wurden, können den folgenden Status haben:

Status	Description
Aktiv	Der Analysator für externen Zugriff überwacht aktiv Ressourcen innerhalb seiner Vertrauen szone. Der Analysator generiert aktiv neue Erkenntnisse und aktualisiert vorhandene Erkenntnisse. Bei Analysatoren für ungenutzten Zugriff überwacht der Analyzer aktiv den ungenutzten Zugriff innerhalb der ausgewählten Organisat ion oder AWS-Konto im angegebenen Nachverfolgungszeitraum. Der Analysator generiert aktiv neue Erkenntnisse und aktualisi ert vorhandene Erkenntnisse.
Erstellen	Die Erstellung des Analysators ist noch im Gange. Der Analysator wird aktiv, sobald die Erstellung abgeschlossen ist.
Disabled	Der Analyzer ist aufgrund einer Aktion des AWS Organizations Administrators deaktiviert. Beispiel: Entfernen des Kontos des Analysato rs als delegierter Administrator für IAM Access Analyzer. Wenn sich der Analysator in einem deaktivierten Zustand befindet, generiert er keine neuen Ergebnisse oder aktualisiert keine vorhandenen Ergebnisse mehr.
Fehlgeschlagen	Die Erstellung des Analysators ist aufgrund eines Konfigurationsproblems fehlgeschlagen. Der Analysator generiert keine Ergebnisse.

Status	Description
	Löschen Sie den Analysator und erstellen Sie einen neuen Analysator.

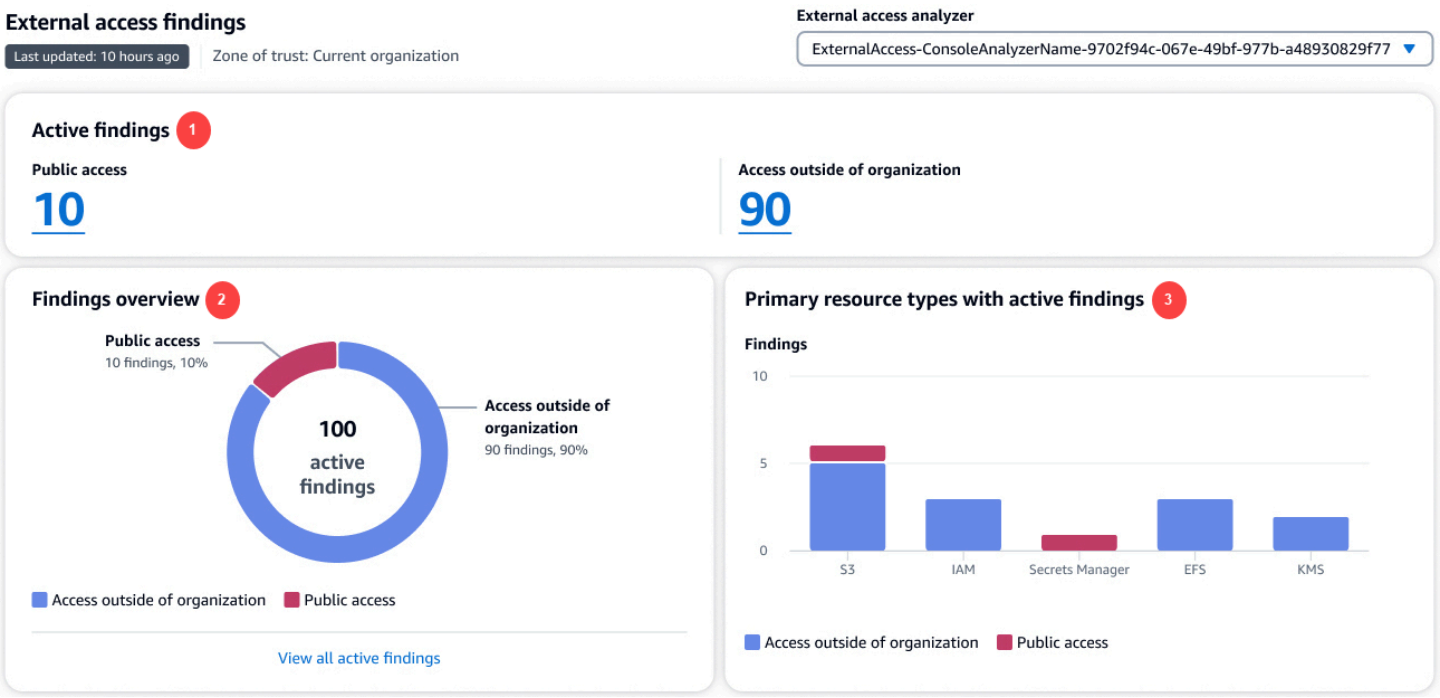
Anzeigen des Dashboards mit den Ergebnissen von IAM Access Analyzer

AWS Identity and Access Management Access Analyzer organisiert externen Zugriff und ungenutzte Zugriffsergebnisse in einem visuellen Übersichts-Dashboard. Das Dashboard hilft Ihnen dabei, sich einen Überblick über die effektive Nutzung von Berechtigungen in großem Umfang zu verschaffen und Konten zu identifizieren, die Ihrer Aufmerksamkeit bedürfen. Sie können das Dashboard verwenden, um die Ergebnisse nach AWS Organisation, Konto und Ergebnisart zu überprüfen.

So zeigen Sie das Übersichts-Dashboard für Analysatoren für externen Zugriff an

> **Note**
>
> Nachdem Sie einen Analysator erstellt oder aktualisiert haben, kann es einige Zeit dauern, bis das Übersichts-Dashboard die Aktualisierungen der Ergebnisse wiedergibt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Auf Analysator zugreifen. Das Fenster Zusammenfassung wird angezeigt.
3. Wählen Sie in der Dropdownliste Analysator für externen Zugriff einen Analysator aus. Eine Zusammenfassung der Ergebnisse für den Analysator wird im Abschnitt Ergebnisse des externen Zugriffs angezeigt.



In der vorhergehenden Abbildung wird das Dashboard mit den Ergebnissen des externen Zugriffs auf der Seite Zusammenfassung angezeigt:

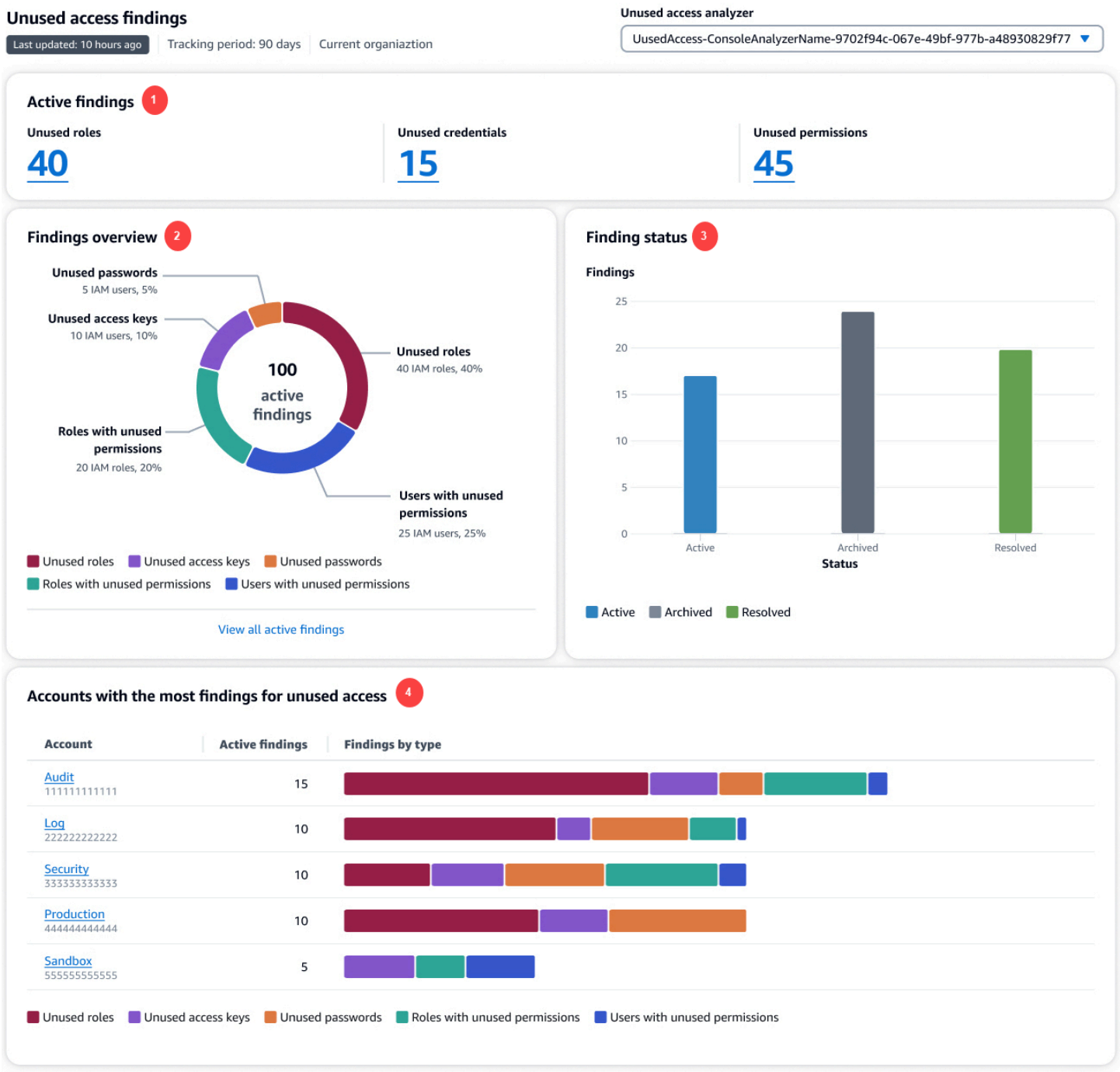
1. Der Abschnitt Aktive Ergebnisse zeigt die Anzahl der aktiven Ergebnisse mit öffentlichem Zugriff sowie die Anzahl der aktiven Ergebnisse, bei denen der Zugriff außerhalb des Kontos oder der Organisation möglich ist. Wählen Sie eine Zahl, um alle aktiven Ergebnisse jedes Typs aufzulisten.
2. Der Abschnitt Überblick über die Ergebnisse enthält eine Aufschlüsselung der Art der aktiven Ergebnisse. Wählen Sie Alle aktiven Ergebnisse anzeigen, um eine vollständige Liste der aktiven Ergebnisse für das Konto oder die Organisation des Analysators anzuzeigen.
3. Der Abschnitt Primäre Ressourcentypen mit aktiven Ergebnissen enthält eine Aufschlüsselung der primären Ressourcentypen mit aktiven Ergebnissen. Diese Informationen helfen Ihnen dabei, die Ergebnisse zunächst für die primären Ressourcen zu priorisieren. Zum Beispiel Amazon S3, DynamoDB und. AWS KMS Dies ist keine vollständige Liste aller Ressourcentypen. Ihr Analysator hat möglicherweise aktive Ergebnisse für Ressourcentypen, die in diesem Abschnitt nicht aufgeführt sind.

So zeigen Sie das Übersichts-Dashboard für Analysatoren für ungenutzten Zugriff an

IAM Access Analyzer erhebt Gebühren für die Analyse des ungenutzten Zugriffs auf der Grundlage der Anzahl der pro Monat analysierten IAM-Rollen und -Benutzer. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

> **Note**
>
> Nachdem Sie einen Analysator erstellt oder aktualisiert haben, kann es einige Zeit dauern, bis das Übersichts-Dashboard die Aktualisierungen der Ergebnisse wiedergibt, je nach Anzahl der Benutzer und Rollen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Auf Analysator zugreifen. Das Fenster Zusammenfassung wird angezeigt.
3. Wählen Sie in der Dropdownliste Analysator für ungenutzten Zugriff einen Analysator aus. Eine Zusammenfassung der Ergebnisse für den Analysator wird im Abschnitt Ergebnisse des ungenutzten Zugriffs angezeigt.



In der vorhergehenden Abbildung wird das Dashboard mit den Ergebnissen des externen Zugriffs auf der Seite Zusammenfassung angezeigt:

1. Der Abschnitt Aktive Ergebnisse enthält die Anzahl der aktiven Ergebnisse für ungenutzte Rollen, ungenutzte Anmeldeinformationen und ungenutzte Berechtigungen in Ihrem Konto oder Ihrer Organisation. Zu den ungenutzten Anmeldedaten gehören sowohl ungenutzte Zugriffsschlüssel als auch ungenutzte Passwörter. Zu den ungenutzten Berechtigungen gehören sowohl Benutzer als

auch Rollen mit ungenutzten Berechtigungen. Wählen Sie eine Zahl, um alle aktiven Ergebnisse jedes Typs aufzulisten.

2. Der Abschnitt Überblick über die Ergebnisse enthält eine Aufschlüsselung der Art der aktiven Ergebnisse. Wählen Sie Alle aktiven Ergebnisse anzeigen, um eine vollständige Liste der aktiven Ergebnisse für das Konto oder die Organisation des Analysators anzuzeigen.
3. Der Abschnitt Ergebnisstatus enthält eine Aufschlüsselung des Status der Ergebnisse (Aktiv, Archiviert und Gelöst) für Ihr Konto oder Ihre Organisation.
4. Der Abschnitt Konten mit den meisten Ergebnissen für ungenutzten Zugriff wird nur angezeigt, wenn sich die ausgewählten Konten Ihres Analysators für ungenutzten Zugriff auf Organisationsebene befinden. Er enthält eine Aufschlüsselung der Konten in Ihrer Organisation mit den aktivsten Ergebnissen. Dies ist keine vollständige Liste aller Konten in Ihrer Organisation. Ihr Analysator hat möglicherweise aktive Ergebnisse für andere Konten, die in diesem Abschnitt nicht aufgeführt sind.

Arbeiten mit Ergebnissen

Ergebnisse zum externen Zugriff

Ergebnisse zum externen Zugriff werden für jede Instance einer Ressource, die außerhalb Ihrer Vertrauenszone freigegeben wird, nur einmal generiert. Jedes Mal, wenn eine ressourcenbasierte Richtlinie geändert wird, analysiert IAM Access Analyzer die Richtlinie. Wenn die aktualisierte Richtlinie eine Ressource freigibt, die bereits mit anderen Berechtigungen oder Bedingungen in einem Ergebnis identifiziert wurde, wird für diese Instance der Ressourcenfreigabe ein neues Ergebnis generiert. Wenn der Zugriff im ersten Ergebnis entfernt wird, erhält dieses Ergebnis den Status Gelöst.

Der Status aller Ergebnisse bleibt Aktiv, bis Sie sie archivieren oder den Zugriff, der das Ergebnis generiert hat, entfernen. Wenn Sie den Zugriff entfernen, wird der Ergebnisstatus auf Gelöst aktualisiert.

> **Note**
>
> Es kann bis zu 30 Minuten dauern, nachdem eine Richtlinie geändert wurde, damit IAM Access Analyzer die Ressource analysiert und das Ergebnis zum externen Zugriff aktualisiert.

Ergebnisse zum ungenutzten Zugriff

Ergebnisse zum ungenutzten Zugriff werden für IAM-Entitäten innerhalb des ausgewählten Kontos oder der Organisation basierend auf der Anzahl von Tagen generiert, die bei der Erstellung des Analysators angegeben wurde. Wenn der Analysator die Entitäten das nächste Mal scannt, wird ein neues Ergebnis generiert, wenn eine der folgenden Bedingungen erfüllt ist:

- Eine Rolle ist für die angegebene Anzahl von Tagen inaktiv.
- Eine ungenutzte Berechtigung, ein ungenutztes Benutzerpasswort oder ein ungenutzter Benutzerzugriffsschlüssel überschreitet die angegebene Anzahl von Tagen.

Sie sollten alle Ergebnisse in Ihrem Konto überprüfen, um festzustellen, ob das Ergebnis zum externen oder ungenutzten Zugriff erwartet und genehmigt wird. Wird der im Ergebnis erkannte externe oder ungenutzte Zugriff erwartet, können Sie das Ergebnis archivieren. Wenn Sie ein Ergebnis archivieren, wird der Status auf Archiviert geändert, und das Ergebnis wird aus der Liste der aktiven Ergebnisse entfernt. Das Ergebnis wird nicht gelöscht. Sie können Ihre archivierten Ergebnisse jederzeit einsehen. Arbeiten Sie alle Ergebnisse in Ihrem Konto durch, bis keine aktiven Ergebnisse mehr vorliegen. Sobald Sie null Ergebnisse erreicht haben, wissen Sie, dass alle neu generierten Ergebnisse im Status Aktiv aus einer kürzlichen Änderung an der Umgebung stammen.

> **Note**
>
> Ungenutzte Zugriffsergebnisse sind nur mit der ListFindingsV2-API-Aktion verfügbar.

Überprüfung der Ergebnisse

Nach der Aktivierung von IAM Access Analyzer müssen Sie im nächsten Schritt alle Ergebnisse überprüfen, um festzustellen, ob der im Ergebnis identifizierte Zugriff beabsichtigt oder unbeabsichtigt ist. Sie können Ergebnisse auch überprüfen, um ähnliche Ergebnisse für beabsichtigten Zugriff zu ermitteln, Dann können Sie eine Archivregel erstellen, um diese Ergebnisse automatisch zu archivieren. Sie können archivierte und gelöste Ergebnisse auch überprüfen.

So überprüfen Sie die Ergebnisse

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Auf Analysator zugreifen.

3. Das Ergebnis-Dashboard wird angezeigt. Wählen Sie die aktiven Ergebnisse für Ihren Analysator für externen oder ungenutzten Zugriff aus.

   Weitere Informationen zum Anzeigen des Ergebnis-Dashboards finden Sie unter Anzeigen des Dashboards mit den Ergebnissen von IAM Access Analyzer.

> **Note**
>
> Ergebnisse werden nur angezeigt, wenn Sie dazu berechtigt sind, Ergebnisse für den Analysator anzuzeigen.

Alle Ergebnisse werden dem Analysator angezeigt. Um andere vom Analysator generierte Ergebnisse anzuzeigen, wählen Sie den entsprechenden Ergebnistyp aus der Dropdownliste Status aus:

- Wählen Sie Active (Aktiv) aus, um alle aktiven Ergebnisse anzuzeigen, die vom Analysator generiert wurden.
- Wählen Sie Archived (Archiviert) aus, um nur die Ergebnisse anzuzeigen, die vom Analysator generiert und archiviert wurden. Weitere Informationen hierzu finden Sie unter Archivieren von Ergebnissen.
- Wählen Sie Resolved (Gelöst) aus, um nur die Ergebnisse anzuzeigen, die vom Analysator generiert und gelöst wurden. Wenn Sie das Problem beheben, weshalb das Ergebnis generiert wurde, wird der Ergebnisstatus auf Gelöst geändert.

  > **Important**
  >
  > Gelöste Ergebnisse werden 90 Tage nach der letzten Aktualisierung des Ergebnisses gelöscht. Aktive und archivierte Ergebnisse werden nur gelöscht, wenn Sie den Analysator löschen, der sie generiert hat.

- Wählen Sie All (Alle) aus, um alle vom Analysator generierten Ergebnisse mit einem beliebigen Status anzuzeigen.

Ergebnisse zum externen Zugriff

Wählen Sie Externer Zugriff und dann den Analysator für externen Zugriff in der Dropdownliste Analysator anzeigen aus. Auf der Seite Ergebnisse für Analysatoren für externen Zugriff werden die folgenden Details zu der freigegebenen Ressource und der Richtlinienanweisung angezeigt, die das Ergebnis generiert hat:

Die ID des Ergebnisses

Die eindeutige ID, die dem Ergebnis zugewiesen ist. Wählen Sie die Ergebnis-ID aus, um zusätzliche Details zu der Ressource und der Richtlinienanweisung anzuzeigen, die das Ergebnis generiert hat.

Ressource

Der Typ und der Teilnahme der Ressource, auf die eine Richtlinie angewendet wurde, die Zugriff auf eine externe Entität außerhalb Ihrer Vertrauenszone gewährt.

Resource owner account (Ressourcenbesitzer-Konto)

Diese Spalte wird nur angezeigt, wenn Sie eine Organisation als Vertrauenszone verwenden. Das Konto in der Organisation, dem die Ressource gehört, die im Ergebnis gemeldet wurde.

Externer Auftraggeber

Der Auftraggeber, außerhalb Ihrer Vertrauenszone, auf den die analysierte Richtlinie Zugriff gewährt. Gültige Werte sind:

- AWS-Konto: Alle Auftraggeber im aufgeführten AWS-Konto mit Berechtigungen des Administrators dieses Kontos können auf die Ressource zugreifen.
- Beliebiger Principal — Alle Principals AWS-Konto , die die in der Spalte „Bedingungen“ angegebenen Bedingungen erfüllen, sind berechtigt, auf die Ressource zuzugreifen. Wenn beispielsweise eine VPC aufgeführt wird, bedeutet dies, dass jeder Auftraggeber in einem beliebigen Konto, das über die Berechtigung zum Zugriff auf die aufgelistete VPC verfügt, auf die Ressource zugreifen kann.
- Kanonischer Benutzer — Alle Principals in der AWS-Konto mit der angegebenen kanonischen Benutzer-ID sind berechtigt, auf die Ressource zuzugreifen.
- IAM role (IAM-Rolle) – Die aufgeführte IAM-Rolle verfügt über die Zugriffsberechtigung für die Ressource.
- IAM user (IAM-Benutzer) – Der aufgeführte IAM-Benutzer hat Zugriffsberechtigung für die Ressource.

Bedingung

Die Bedingung der Richtlinienanweisung, die den Zugriff gewährt. Wenn beispielsweise das Feld Condition (Bedingung) den Wert Source VPC (Quell-VPC) enthält, bedeutet dies, dass die Ressource für einen Auftraggeber mit Zugriff auf die aufgeführte VPC freigegeben wird. Die Bedingungen können global oder spezifisch für den Service sein. Globale Bedingungsschlüssel haben das Präfix `aws:`.

Shared through (Freigegeben durch)

Das Feld Shared through (Freigegeben durch) gibt an, wie der Zugriff gewährt wird, der das Ergebnis generiert hat. Gültige Werte sind:

- Bucket-Richtlinie – Dem Amazon S3 Bucket angefügte Bucket-Richtlinie.
- Zugriffssteuerungsliste – Die Zugriffskontrollliste (Access Control List, ACL), die dem Amazon S3-Bucket zugeordnet ist.
- Zugriffspunkt – Ein Zugriffspunkt oder Zugriffspunkt mit mehreren Regionen, der dem Amazon S3 Bucket zugeordnet ist. Der ARN des Zugriffspunkts wird in den Details unter Findings (Ergebnisse) angezeigt.

Zugriffsebene

Die Zugriffsebene, die der externen Entität durch die Aktionen in der ressourcenbasierten Richtlinie gewährt wird. Sehen Sie sich die Details der Suche an, um weitere Informationen zu erhalten. Zu den Werte der Zugriffsebene gehören die folgenden:

- List (Aufführen) – die Berechtigung zum Auflisten von Ressourcen innerhalb des Services, um zu bestimmen, ob ein Objekt vorhanden ist. Aktionen mit dieser Zugriffsebene können Objekte auflisten, aber nicht die Inhalte einer Ressource sehen.
- Read (Lesen) – die Berechtigung zum Lesen, jedoch nicht zum Bearbeiten der Inhalte und Attribute der Ressourcen innerhalb des Services.
- Write (Schreiben) – die Berechtigung zum Erstellen, Löschen oder Ändern von Ressourcen innerhalb des Services.
- Permissions (Berechtigungen) – die Berechtigung zum Erteilen oder Ändern von Ressourcenberechtigungen innerhalb des Services.
- Tagging (Markieren) – die Berechtigung zum Ausführen von Aktionen, die nur den Status der Ressourcen-Tags ändern.

Aktualisiert

Ein Zeitstempel für die letzte Aktualisierung des Ergebnisstatus oder die Uhrzeit und das Datum, zu dem das Ergebnis generiert wurde, sofern keine Aktualisierungen vorgenommen wurden.

> **Note**
>
> Es kann nach Änderung einer Richtlinie bis zu 30 Minuten dauern, bis die Ressource erneut von IAM Access Analyzer analysiert und das Ergebnis aktualisiert wird.

Status

Der Status des Ergebnisses, entweder Active (Aktiv), Archived (Archiviert) oder Resolved (Gelöst).

Ergebnisse zum ungenutzten Zugriff

IAM Access Analyzer erhebt Gebühren für die Analyse des ungenutzten Zugriffs auf der Grundlage der Anzahl der pro Monat analysierten IAM-Rollen und -Benutzer. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

Wählen Sie Ungenutzter Zugriff und dann den Analysator für ungenutzten Zugriff in der Dropdownliste Analysator anzeigen aus. Auf der Seite Ergebnisse für Analysatoren für ungenutzten Zugriff werden die folgenden Details über die IAM-Entität angezeigt, die das Ergebnis generiert hat:

Die ID des Ergebnisses

Die eindeutige ID, die dem Ergebnis zugewiesen ist. Wählen Sie die Ergebnis-ID aus, um zusätzliche Details zu der IAM-Entität anzuzeigen, die das Ergebnis generiert hat.

Ergebnistyp

Die Art der Suche nach ungenutztem Zugriff: Ungenutzter Zugriffsschlüssel, Ungenutztes Passwort, Ungenutzte Berechtigung oder Ungenutzte Rolle.

IAM-Entität

Die im Ergebnis gemeldete IAM-Entität. Dabei kann es sich um einen IAM-Benutzer oder -Rolle handeln.

AWS-Konto ID (ID)

Diese Spalte wird nur angezeigt, wenn Sie den Analyzer für alle Mitglieder der Organisation eingerichtet haben. AWS-Konten AWS-Konto In der Organisation, der die IAM-Entität gehört, wurde im Ergebnis angegeben.

Letzte Aktualisierung

Das letzte Mal, dass die im Ergebnis gemeldete IAM-Entität aktualisiert wurde, oder wann die Entität erstellt wurde, wenn keine Aktualisierungen vorgenommen wurden.

Status

Der Status des Ergebnisses (Aktiv, Archiviert oder Gelöst).

Filtern von Ergebnissen

Mit der Standardfilterung werden auf der Ergebnisseite alle aktiven Ergebnisse angezeigt. Um aktive Ergebnisse anzuzeigen, wählen Sie den Status Aktiv aus der Dropdownliste Status aus. Um archivierte Ergebnisse anzuzeigen, wählen Sie den Status Archiviert aus der Dropdownliste Status aus. Wenn Sie IAM Access Analyzer zum ersten Mal verwenden, liegen keine archivierten Ergebnisse vor.

Verwenden Sie Filter, um nur die Ergebnisse anzuzeigen, die den angegebenen Eigenschaftskriterien entsprechen. Um einen Filter zu erstellen, wählen Sie die Eigenschaft aus, nach der gefiltert werden soll. Wählen Sie dann, ob die Eigenschaft gleich ist oder einen Wert enthält, und geben Sie dann einen Eigenschaftswert ein, nach dem gefiltert werden soll, oder wählen Sie ihn aus. Um beispielsweise einen Filter zu erstellen, der nur Ergebnisse für ein bestimmtes Objekt anzeigt AWS-Konto, wählen Sie AWS Konto für die Immobilie aus, wählen Sie dann AWS Konto = und geben Sie dann die Kontonummer für AWS-Konto das Objekt ein, für das Sie Ergebnisse anzeigen möchten.

Eine Liste der Filterschlüssel, mit denen Sie eine Archivregel erstellen oder aktualisieren können, finden Sie unter Filterschlüssel für IAM Access Analyzer.

Filtern von Ergebnissen des externen Zugriffs

S filtern Sie Ergebnisse des externen Zugriffs

1. Wählen Sie Externer Zugriff und dann den Analysator in der Dropdownliste Analysator anzeigen aus.

2. Wählen Sie das Suchfeld, um eine Liste der verfügbaren Eigenschaften anzuzeigen.
3. Wählen Sie die Eigenschaft aus, die zum Filtern der angezeigten Ergebnisse verwendet werden soll.
4. Wählen Sie den entsprechenden Wert für die Eigenschaft. Nur Ergebnisse mit diesem Wert im Ergebnis werden angezeigt.

   Wählen Sie beispielsweise Ressource als Eigenschaft aus, wählen Sie dann Ressource : aus, geben Sie den Namen eines Buckets teilweise oder vollständig ein, und drücken Sie dann die Eingabetaste. Es werden nur die den Filterkriterien entsprechenden Ergebnisse für den Bucket angezeigt. Um einen Filter zu erstellen, der nur Ergebnisse für Ressourcen anzeigt, die öffentlichen Zugriff zulassen, können Sie die Eigenschaft Öffentlicher Zugriff, dann Öffentlicher Zugriff: und dann Öffentlicher Zugriff: true auswählen.

Sie können zusätzliche Eigenschaften hinzufügen, um die angezeigten Ergebnisse weiter zu filtern. Wenn Sie zusätzliche Eigenschaften hinzufügen, werden nur Ergebnisse angezeigt, die allen Bedingungen im Filter entsprechen. Das Definieren eines Filters zur Anzeige von Ergebnissen, die einer Eigenschaft ODER einer anderen Eigenschaft entsprechen, wird nicht unterstützt. Wählen Sie Filter löschen, um alle von Ihnen definierten Filter zu löschen und alle Ergebnisse mit dem angegebenen Status für Ihren Analysator anzuzeigen.

Einige Felder werden nur angezeigt, wenn Sie Ergebnisse für einen Analysator anzeigen, dessen Vertrauenszone eine Organisation ist.

Für die Definition von Filtern stehen folgende Eigenschaften zur Verfügung:

- Public access (Öffentlicher Zugriff) – Um nach Ressourcen zu filtern, die öffentlichen Zugriff ermöglichen, filtern Sie nach Public access (Öffentlicher Zugriff) und wählen dann Public access: true (Öffentlicher Zugriff: true).
- Resource (Ressource) – Geben Sie den Namen der Ressource ganz oder teilweise ein, um nach der Ressource zu filtern.
- Resource Type (Ressourcentyp) – Wählen Sie den Typ aus der angezeigten Liste aus, um nach Ressourcentyp zu filtern.
- Konto des Ressourceneigentümers – Verwenden Sie diese Eigenschaft, um nach dem Konto in der Organisation zu filtern, dem die im Ergebnis gemeldete Ressource gehört.
- AWS Konto — Verwenden Sie diese Eigenschaft, um nach Personen zu filtern AWS-Konto , denen im Hauptbereich einer Richtlinienerklärung Zugriff gewährt wurde. Geben Sie zum Filtern AWS-

Konto die gesamte oder einen Teil der 12-stelligen AWS-Konto ID oder den gesamten oder einen Teil des vollständigen Konto-ARN des externen AWS Benutzers oder der Rolle ein, der Zugriff auf Ressourcen im aktuellen Konto hat.

- Kanonischer Benutzer – Zum Filtern nach kanonischen Benutzern geben Sie die kanonische Benutzer-ID ein, wie sie für Amazon-S3-Buckets definiert ist. Weitere Informationen finden Sie unter AWS -Kontenkennungen.
- Federated User (Verbundener Benutzer) – Geben Sie den ARN der Verbundidentität ganz oder teilweise ein, um nach einem Verbundbenutzer zu filtern. Weitere Informationen finden Sie unter Identitätsanbieter und Verbund.
- Ergebnis-ID – Um nach der Ergebnis-ID zu filtern, geben Sie die Ergebnis-ID ganz oder teilweise ein.
- Principal-ARN — Verwenden Sie diese Eigenschaft, um nach dem ARN des Prinzipals (IAM-Benutzer, Rolle oder Gruppe) zu filtern, der in einem aws: PrincipalArn -Bedingungsschlüssel verwendet wird. Um nach Principal-ARN zu filtern, geben Sie den gesamten oder einen Teil des ARN des IAM-Benutzers, der Rolle oder der Gruppe von einer externen Person ein, die in einem Ergebnis AWS-Konto gemeldet wurde.
- Prinzipal-OrgID zum Filtern nach Prinzipal-OrgID geben Sie die Organisations-ID ganz oder teilweise ein, die den externen Prinzipalen zugewiesen ist, die der als Bedingung im Ergebnis angegebenen AWS -Organisation angehören. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.
- Principal OrgPaths — Um nach Principal zu filtern OrgPaths, geben Sie als Bedingung in der Richtlinie die gesamte oder einen Teil der ID für die AWS Organisation oder Organisationseinheit (OU) ein, die den Zugriff auf alle externen Prinzipale ermöglicht, die Kontomitglieder der angegebenen Organisation oder OU sind. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.
- Quellkonto — Um nach dem Quellkonto zu filtern, geben Sie die AWS-Konto ID, die den Ressourcen zugeordnet ist, ganz oder teilweise ein, wie sie in einigen dienstübergreifenden Berechtigungen unter verwendet wird. AWS Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.
- Source ARN (Quell-ARN) – Sie können nach dem Quell-ARN filtern, indem Sie den ARN, der als Bedingung im Ergebnis angegeben wurde, ganz oder teilweise eingeben. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.
- Source IP (Quell-IP) – Um nach der Quell-IP zu filtern, geben Sie die IP-Adresse, die externen Entitäten bei Verwendung der angegebenen IP-Adresse den Zugriff auf Ressourcen im aktuellen

Konto ermöglicht, ganz oder teilweise ein. Weitere Informationen finden Sie unter Globale AWS - Bedingungskontextschlüssel.

- Source VPC (Quell-VPC) – Wenn Sie nach Quell-VPC filtern möchten, geben Sie die VPC-ID, die externen Entitäten bei Verwendung der angegebenen VPC den Zugriff auf Ressourcen im aktuellen Konto ermöglicht, ganz oder teilweise ein. Weitere Informationen finden Sie unter Globale AWS - Bedingungskontextschlüssel.
- Quell-OrgID — Um nach Quell-OrgID zu filtern, geben Sie die Organisations-ID, die den Ressourcen zugeordnet ist, ganz oder teilweise ein, wie sie in einigen dienstübergreifenden Berechtigungen unter verwendet wird. AWS Weitere Informationen finden Sie unter Globale AWS - Bedingungskontextschlüssel.
- Quelle OrgPaths — Um nach Quelle zu filtern OrgPaths, geben Sie die gesamte oder einen Teil der Organisationseinheit (OU) ein, die den Ressourcen zugeordnet ist, wie sie in einigen dienstübergreifenden Berechtigungen unter verwendet wird. AWS Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.
- Benutzer-ID — Um nach Benutzer-ID zu filtern, geben Sie die gesamte oder einen Teil der Benutzer-ID des IAM-Benutzers von einem externen Benutzer ein, der Zugriff auf AWS-Konto die Ressource im aktuellen Konto hat. Weitere Informationen finden Sie unter Globale AWS - Bedingungskontextschlüssel.
- KMS-Schlüssel-ID — Um nach der KMS-Schlüssel-ID zu filtern, geben Sie die gesamte oder einen Teil der Schlüssel-ID für den KMS-Schlüssel ein, der als Bedingung für den AWS KMS-verschlüsselten Amazon S3 S3-Objektzugriff in Ihrem aktuellen Konto angegeben ist.
- Google Audience (Google Audience) – Um nach Google Audience zu filtern, geben Sie die Google-Anwendungs-ID, die als Bedingung für den IAM-Rollenzugriff in Ihrem aktuellen Konto angegeben ist, ganz oder teilweise ein. Weitere Informationen finden Sie unter IAM- und AWS STS Bedingungskontextschlüssel.
- Cognito-Zielgruppe – Zum Filtern nach Amazon-Cognito-Zielgruppe geben Sie die ID des Amazon-Cognito-Identitätspools, die als Bedingung für den Zugriff auf IAM-Rollen in Ihrem aktuellen Konto angegeben ist, ganz oder teilweise ein. Weitere Informationen finden Sie unter IAM- und AWS STS Bedingungskontextschlüssel.
- Anruferkonto — Die AWS-Konto ID des Kontos, das die aufrufende Entität besitzt oder enthält, z. B. eine IAM-Rolle, ein Benutzer oder ein Kontostammbenutzer. Dies wird von Diensten verwendet, die anrufen. AWS KMS Um nach Aufruferkonto zu filtern, geben Sie die AWS-Konto -ID ganz oder teilweise ein.
- Facebook App ID (Facebook-App-ID) – Wenn Sie nach der Facebook-App-ID filtern möchten, geben Sie die Facebook-Anwendungs-ID (oder Website-ID), die als Bedingung angegeben ist,

ganz oder teilweise ein. Damit wird der Zugriff auf eine IAM-Rolle in Ihrem aktuellen Konto bei Anmeldung mit dem Facebook-Verbund ermöglicht. Weitere Informationen finden Sie im Abschnitt id unter IAM und AWS STS -Bedingungskontextschlüssel.

- Amazon App ID (Amazon-App-ID) – Wenn Sie nach Amazon App-ID filtern möchten, geben Sie die Amazon-App-ID (oder die Website-ID), die als Bedingung angegeben ist, ganz oder teilweise ein, um Login with Amazon den Verbundzugriff auf eine IAM-Rolle in Ihrem aktuellen Konto zu ermöglichen. Weitere Informationen finden Sie im Abschnitt id unter IAM und AWS STS -Bedingungskontextschlüssel.
- Lambda Event Source Token (Lambda-Ereignisquellen-Token) – Sie können nach Lambda-Ereignisquellen-Token filtern, das mit Alexa-Integrationen übergeben wurde, indem Sie die Token-Zeichenfolge ganz oder teilweise eingeben.

Filtern von Ergebnissen zum ungenutzten Zugriff

So filtern Sie die Ergebnisse zum ungenutzten Zugriff

1. Wählen Sie Ungenutzter Zugriff und dann den Analysator in der Dropdownliste Analysator anzeigen aus.
2. Wählen Sie das Suchfeld, um eine Liste der verfügbaren Eigenschaften anzuzeigen.
3. Wählen Sie die Eigenschaft aus, die zum Filtern der angezeigten Ergebnisse verwendet werden soll.
4. Wählen Sie den entsprechenden Wert für die Eigenschaft. Nur Ergebnisse mit diesem Wert im Ergebnis werden angezeigt.

   Wählen Sie beispielsweise Findings type als Eigenschaft, dann Findings type = und anschließend Findings type = UnuseDiamRole. Nur Ergebnisse mit dem Typ UnuseDiamRole werden angezeigt.

Sie können zusätzliche Eigenschaften hinzufügen, um die angezeigten Ergebnisse weiter zu filtern. Wenn Sie zusätzliche Eigenschaften hinzufügen, werden nur Ergebnisse angezeigt, die allen Bedingungen im Filter entsprechen. Das Definieren eines Filters zur Anzeige von Ergebnissen, die einer Eigenschaft ODER einer anderen Eigenschaft entsprechen, wird nicht unterstützt. Wählen Sie Filter löschen, um alle von Ihnen definierten Filter zu löschen und alle Ergebnisse mit dem angegebenen Status für Ihren Analysator anzuzeigen.

Die folgenden Felder werden nur angezeigt, wenn Sie Ergebnisse für einen Analysator anzeigen, der ungenutzten Zugriff überwacht:

- Art der Ergebnisse — Um nach der Art der Ergebnisse zu filtern, filtern Sie nach der Art der Ergebnisse und wählen Sie dann die Art des Ergebnisses aus.
- Resource (Ressource) – Geben Sie den Namen der Ressource ganz oder teilweise ein, um nach der Ressource zu filtern.
- Resource Type (Ressourcentyp) – Wählen Sie den Typ aus der angezeigten Liste aus, um nach Ressourcentyp zu filtern.
- Konto des Ressourceneigentümers – Verwenden Sie diese Eigenschaft, um nach dem Konto in der Organisation zu filtern, dem die im Ergebnis gemeldete Ressource gehört.
- Such-ID — Um nach der Such-ID zu filtern, geben Sie die gesamte Such-ID oder einen Teil davon ein.

Archivieren von Ergebnissen

Wenn Sie ein Ergebnis für einen beabsichtigten Zugriff auf eine Ressource erhalten, können Sie das Ergebnis archivieren. Zum Beispiel ein Ergebnis für einen externen Zugriff auf eine IAM-Rolle, die von mehreren Benutzern für genehmigte Workflows verwendet wird, oder ein Ergebnis für einen ungenutzten Zugriff auf einen Zugriffsschlüssel, der möglicherweise noch benötigt wird. Wenn Sie ein Ergebnis archivieren, wird es aus der Liste der aktiven Ergebnisse gelöscht. Archivierte Ergebnisse werden nicht gelöscht. Sie können die Seite Ergebnisse filtern, um die archivierten Ergebnisse anzuzeigen und die Archivierung jederzeit wieder aufzuheben.

So archivieren Sie Ergebnisse auf der Seite Ergebnisse

1. Aktivieren Sie das Kontrollkästchen neben mindestens einem zu archivierenden Ergebnis.
2. Klicken Sie auf Aktionen und auf Archivieren.

   Eine Bestätigung wird oben auf dem Bildschirm angezeigt.

So archivieren Sie Ergebnisse auf der Seite Ergebnisdetails

1. Wählen Sie die Finding ID (Ergebnis-ID) für das zu archivierende Ergebnis.
2. Wählen Sie Archiv.

Eine Bestätigung wird oben auf dem Bildschirm angezeigt.

Wenn Sie die Archivierung von Ergebnissen aufheben möchten, wiederholen Sie die vorherigen Schritte, wählen Sie jedoch Unarchive (Archivierung aufheben) statt Archive (Archivieren). Wenn Sie die Archivierung eines Ergebnisses aufheben, wird der Status auf „Aktiv“ festgelegt.

Lösen von Ergebnissen

Ergebnisse zum externen Zugriff

Sie können Ergebnisse zum externen Zugriff, die durch unbeabsichtigten Zugriff generiert wurden, lösen, indem Sie die Richtlinienanweisung so ändern, dass die Berechtigungen entfernt werden, die den Zugriff auf die identifizierte Ressource ermöglichen. Bei Ergebnissen für Amazon-S3-Buckets beispielsweise können Sie die Berechtigungen für den Bucket mit der Amazon-S3-Konsole konfigurieren. Bei IAM-Rollen verwenden Sie die IAM-Konsole zum Ändern der Vertrauensrichtlinie für die aufgeführte IAM-Rolle. Verwenden Sie die Konsole für die anderen unterstützten Ressourcen, um die Richtlinienanweisungen zu ändern, die zu einem generierten Ergebnis geführt haben.

Nachdem Sie eine Änderung vorgenommen haben, um ein Ergebnis zum externen Zugriff aufzulösen, (z. B. das Ändern einer Richtlinie, die auf eine IAM-Rolle angewendet wurde) überprüft IAM Access Analyzer die Ressource erneut. Erfolgt keine weitere Freigabe der Ressource außerhalb Ihrer Vertrauenszone mehr, wird der Status des Ergebnisses auf „Gelöst“ geändert. Das Ergebnis wird nicht mehr in der Liste „Aktive Ergebnisse“, sondern in der Liste „Gelöste Ergebnisse“ angezeigt.

> **Note**
>
> Dies gilt nicht für Fehlerergebnisse. Wenn IAM Access Analyzer eine Ressource nicht analysieren kann, generiert es eine Fehlermeldung. Wenn Sie das Problem beheben, das IAM Access Analyzer daran gehindert hat, die Ressource zu analysieren, wird das Ergebnis vollständig entfernt und nicht in ein behobenes Ergebnis geändert.

Wenn die von Ihnen vorgenommenen Änderungen dazu geführt haben, dass die Ressource außerhalb Ihrer Vertrauenszone geteilt wird, allerdings auf eine andere Weise, z. B. mit einem anderen Auftraggeber oder einer anderen Berechtigung, generiert IAM Access Analyzer ein neues aktives Ergebnis.

> **Note**
>
> Es kann nach Änderung einer Richtlinie bis zu 30 Minuten dauern, bis die Ressource erneut von IAM Access Analyzer analysiert und das Ergebnis aktualisiert wird. Gelöste Ergebnisse werden 90 Tage nach der letzten Aktualisierung des Ergebnisstatus gelöscht.

Ergebnisse zum ungenutzten Zugriff

Für ungenutzte Access Analyzer-Ergebnisse bietet IAM Access Analyzer je nach Art des Ergebnisses empfohlene Schritte zur Behebung der Ergebnisse.

Nachdem Sie eine Änderung zur Auflösung eines Ergebnisses für ungenutzten Zugriff vorgenommen haben, wird der Status des Ergebnisses in Gelöst geändert, wenn der Analysator für ungenutzten Zugriff das nächste Mal ausgeführt wird. Das Ergebnis wird nicht mehr in der Liste der aktiven Ergebnisse angezeigt, sondern in der Liste der gelösten Ergebnisse. Wenn Sie eine Änderung vornehmen, mit der ein Ergebnis zum ungenutzten Zugriff nur teilweise behoben wird, wird der vorhandene Ergebnis zu Gelöst geändert, es wird jedoch ein neues Ergebnis generiert. Sie entfernen beispielsweise nur einige der ungenutzten Berechtigungen in einem Ergebnis, aber nicht alle.

IAM Access Analyzer erhebt Gebühren für die Analyse des ungenutzten Zugriffs auf der Grundlage der Anzahl der pro Monat analysierten IAM-Rollen und -Benutzer. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

Lösung ungenutzter Berechtigungsergebnisse

Bei Ergebnissen ungenutzter Berechtigungen kann IAM Access Analyzer Richtlinien empfehlen, die von einem IAM-Benutzer oder einer IAM-Rolle entfernt werden sollen, und neue Richtlinien bereitstellen, um bestehende Berechtigungsrichtlinien zu ersetzen. Richtlinienempfehlungen werden für die folgenden Szenarien nicht unterstützt:

- Die Suche nach ungenutzten Berechtigungen bezieht sich auf einen IAM-Benutzer, der sich in einer Benutzergruppe befindet.
- Die Suche nach ungenutzten Berechtigungen bezieht sich auf eine IAM-Rolle für IAM Identity Center.
- Für die Suche nach ungenutzten Berechtigungen gibt es bereits eine Berechtigungsrichtlinie, die das `notAction` Element enthält.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Ungenutzter Zugriff aus.
3. Wählen Sie ein Ergebnis mit dem Suchtyp „Ungenutzte Berechtigungen“ aus.
4. Wenn in der Spalte Empfohlene Richtlinie Richtlinien aufgeführt sind, wählen Sie im Abschnitt Empfehlungen die Option Richtlinie in der Vorschau aus, um die bestehende Richtlinie mit der empfohlenen Richtlinie anzuzeigen, die die bestehende Richtlinie ersetzen soll. Wenn es mehrere empfohlene Richtlinien gibt, können Sie „Nächste Richtlinie“ und „Vorherige Richtlinie“ auswählen, um alle vorhandenen und empfohlenen Richtlinien anzuzeigen.
5. Wählen Sie JSON herunterladen, um eine ZIP-Datei mit JSON-Dateien aller empfohlenen Richtlinien herunterzuladen.
6. Erstellen Sie die empfohlenen Richtlinien und fügen Sie sie dem IAM-Benutzer oder der IAM-Rolle hinzu. Weitere Informationen finden Sie unter Ändern der Berechtigungen für einen Benutzer (Konsole) und Ändern einer Rollenberechtigungsrichtlinie (Konsole).
7. Entfernen Sie die Richtlinien, die in der Spalte „Bestehende Berechtigungsrichtlinie“ aufgeführt sind, aus dem IAM-Benutzer oder der IAM-Rolle. Weitere Informationen finden Sie unter Entfernen von Berechtigungen von einem Benutzer (Konsole) und Ändern einer Rollenberechtigungsrichtlinie (Konsole).

Behebung ungenutzter Rollenergebnisse

Bei Ergebnissen ungenutzter Rollen empfiehlt IAM Access Analyzer, die ungenutzte IAM-Rolle zu löschen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Ungenutzter Zugriff.
3. Wählen Sie ein Ergebnis mit dem Suchtyp „Unbenutzt“ aus.
4. Sehen Sie sich im Abschnitt Empfehlungen die Details der IAM-Rolle an.
5. Löschen Sie die IAM-Rolle. Weitere Informationen finden Sie unter Löschen einer IAM-Rolle (Konsole).

Behebung ungenutzter Zugriffsschlüsselergebnisse

Bei Ergebnissen ungenutzter Zugriffsschlüssel empfiehlt IAM Access Analyzer, den ungenutzten Zugriffsschlüssel zu deaktivieren oder zu löschen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Unbenutzter Zugriff.
3. Wählen Sie ein Ergebnis mit dem Suchtyp „Unbenutzte Zugangsschlüssel“.
4. Überprüfen Sie im Abschnitt Empfehlungen die Details des Zugriffsschlüssels.
5. Deaktivieren oder löschen Sie den Zugriffsschlüssel. Weitere Informationen finden Sie unter Zugriffsschlüssel verwalten (Konsole).

Behebung ungenutzter Kennwortergebnisse

Bei Ergebnissen ungenutzter Passwörter empfiehlt IAM Access Analyzer, das unbenutzte Passwort für den IAM-Benutzer zu löschen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Unbenutzter Zugriff.
3. Wählen Sie ein Ergebnis mit dem Suchtyp Unbenutztes Passwort.
4. Überprüfen Sie im Abschnitt Empfehlungen die Details des IAM-Benutzers.
5. Löschen Sie das Passwort für den IAM-Benutzer. Weitere Informationen finden Sie unter Ein IAM-Benutzerkennwort erstellen, ändern oder löschen (Konsole).

Ressourcentypen für IAM Access Analyzer für externen Zugriff

Für externe Zugriffsanalysatoren analysiert IAM Access Analyzer die ressourcenbasierten Richtlinien, die auf AWS Ressourcen in der Region angewendet werden, in der Sie IAM Access Analyzer aktiviert haben. Es werden nur ressourcenbasierte Richtlinien analysiert. Überprüfen Sie die Informationen zu jeder Ressource, um zu erfahren, wie IAM Access Analyzer Ergebnisse für jeden Ressourcentyp generiert.

> **Note**
>
> Die aufgeführten unterstützten Ressourcentypen gelten für Analysatoren für externen Zugriff. Analysatoren für ungenutzten Zugriff unterstützen nur IAM-Benutzer und -Rollen. Weitere Informationen finden Sie unter Arbeiten mit Ergebnissen.

Unterstützte Ressourcentypen für den externen Zugriff:

- Amazon-Simple-Storage-Service-Buckets
- Amazon-Simple-Storage-Service-Verzeichnis-Buckets
- AWS Identity and Access Management Rollen
- AWS Key Management Service Schlüssel
- AWS Lambda Funktionen und Schichten
- Amazon Simple Queue Service Warteschlangen als Ziele
- AWS Secrets Manager Geheimnisse
- Amazon Simple Notification Service-Themen
- Volume-Snapshots von Amazon Elastic Block Store
- DB-Snapshots von Amazon Relational Database Service
- Snapshots von Service-DB-Clustern von Amazon Relational Database
- Repositories der Amazon Elastic Container Registry
- Dateisysteme des Amazon Elastic File System
- Amazon DynamoDB DynamoDB-Streams
- Amazon-DynamoDB-Tabellen

Amazon-Simple-Storage-Service-Buckets

Wenn IAM Access Analyzer Amazon-S3-Buckets analysiert, generiert es ein Ergebnis, wenn eine Amazon-S3-Bucket-Richtlinie, ACL oder ein Zugriffspunkt, einschließlich eines Zugriffspunkts mit mehreren Regionen, der auf einen Bucket angewendet wird, einer externen Entität Zugriff gewährt. Eine externe Entität ist ein Prinzipal oder eine andere Entität, mit der Sie einen Filter erstellen können, der sich nicht innerhalb Ihrer Vertrauenszone befindet. Wenn beispielsweise eine Bucket-Richtlinie Zugriff auf ein anderes Konto gewährt oder öffentlichen Zugriff zulässt, generiert IAM Access Analyzer ein Ergebnis. Wenn Sie Block Public Access (Öffentlichen Zugriff blockieren) aktivieren, können Sie den Zugriff auf Konto- oder Bucket-Ebene blockieren.

> **Note**
>
> IAM Access Analyzer analysiert die Zugriffspunktrichtlinie, die an kontoübergreifende Zugriffspunkte angehängt ist, nicht, da sich der Zugriffspunkt und seine Richtlinie außerhalb des Analyzer-Kontos befinden. IAM Access Analyzer generiert ein öffentliches Ergebnis, wenn ein Bucket den Zugriff an einen kontoübergreifenden Zugriffspunkt delegiert und „Block Public Access (Öffentlichen Zugriff blockieren)“ für den Bucket oder das Konto nicht aktiviert

ist. Wenn Sie „Block Public Access (Öffentlichen Zugriff blockieren)“ aktivieren, wird die öffentliche Feststellung aufgelöst und IAM Access Analyzer generiert eine kontoübergreifende Feststellung für den kontoübergreifenden Zugriffspunkt.

Die Einstellungen für Amazon S3 Block Public Access haben Vorrang vor den Bucket-Richtlinien, die auf den Bucket angewendet werden. Die Einstellungen überschreiben auch die Zugriffspunkt-Richtlinien, die auf die Zugriffspunkte des Buckets angewendet werden. IAM Access Analyzer analysiert die Block Public Access-Einstellungen auf Bucket-Ebene, wenn sich eine Richtlinie ändert. Allerdings werden die Block Public Access-Einstellungen auf Kontoebene nur einmal alle 6 Stunden ausgewertet. Dies bedeutet, dass IAM Access Analyzer möglicherweise bis zu 6 Stunden lang kein Ergebnis für den öffentlichen Zugriff auf einen Bucket generiert oder auflöst. Wenn Sie beispielsweise eine Bucket-Richtlinie haben, die öffentlichen Zugriff zulässt, generiert IAM Access Analyzer ein Ergebnis für diesen Zugriff. Falls Sie „Block Public Access (Öffentlichen Zugriff blockieren)“ aktivieren, um den gesamten öffentlichen Zugriff auf den Bucket auf Kontoebene zu blockieren, löst IAM Access Analyzer das Ergebnis für die Bucket-Richtlinie bis zu 6 Stunden lang nicht auf, obwohl der gesamte öffentliche Zugriff auf den Bucket blockiert ist. Die Lösung von öffentlichen Feststellungen für kontoübergreifende Zugriffspunkte kann ebenfalls bis zu 6 Stunden dauern, sobald Sie auf Kontoebene die Option „Block Public Access (Öffentlichen Zugriff blockieren)“ aktiviert haben.

Für einen Zugriffspunkt für mehrere Regionen verwendet IAM Access Analyzer eine festgelegte Richtlinie zum Generieren von Ergebnissen. IAM Access Analyzer wertet alle 6 Stunden Änderungen an Zugriffspunkten mit mehreren Regionen aus. Dies bedeutet, dass IAM Access Analyzer bis zu 6 Stunden lang keine Ergebnisse erzeugt oder auflöst, selbst wenn Sie einen Zugangspunkt mit mehreren Regionen erstellen oder löschen oder die Richtlinie dafür aktualisieren.

Amazon-Simple-Storage-Service-Verzeichnis-Buckets

Amazon-S3-Verzeichnis-Buckets verwenden die Speicherklasse „Amazon S3 Express One“, die für anspruchsvolle Workloads bzw. Anwendungen empfohlen wird. Für Amazon-S3-Verzeichnis-Buckets analysiert IAM Access Analyzer die Verzeichnis-Bucket-Richtlinien, einschließlich Bedingungsanweisungen in einer Richtlinie, die einer externen Entität Zugriff auf ein Verzeichnis-Bucket gewähren. Weitere Informationen über Amazon-S3-Verzeichnis-Buckets finden Sie unter Verzeichnis-Buckets im Benutzerhandbuch für Amazon Simple Storage Service.

AWS Identity and Access Management Rollen

Für IAM-Rollen analysiert IAM Access Analyzer Vertrauensrichtlinien. In einer Rollenvertrauensrichtlinie definieren Sie die Auftraggeber, denen Sie bei einer Übernahme der Rolle

vertrauen. Eine Rollenvertrauensrichtlinie ist eine erforderliche ressourcenbasierte Richtlinie die einer Rolle in IAM angefügt ist. IAM Access Analyzer generiert Ergebnisse für Rollen innerhalb der Vertrauenszone, auf die von einer externen Entität zugegriffen werden kann, die sich außerhalb Ihrer Vertrauenszone befindet.

> **Note**
>
> Eine IAM-Rolle ist eine globale Ressource. Wenn eine Rollen-Vertrauensrichtlinie Zugriff auf eine externe Entität gewährt, generiert IAM Access Analyzer ein Ergebnis in jeder aktivierten Region.

AWS Key Management Service Schlüssel

Denn AWS KMS keys IAM Access Analyzer analysiert die wichtigsten Richtlinien und Zuschüsse, die auf einen Schlüssel angewendet werden. IAM Access Analyzer generiert ein Ergebnis, wenn eine Schlüsselrichtlinie oder -gewährung einer externen Entität den Zugriff auf den Schlüssel gewährt. Wenn Sie beispielsweise den CallerAccount Bedingungsschlüssel kms: in einer Richtlinienanweisung verwenden, um allen Benutzern in einem bestimmten AWS Konto Zugriff zu gewähren, und Sie ein anderes Konto als das aktuelle Konto (die Vertrauenszone für den aktuellen Analyzer) angeben, generiert IAM Access Analyzer ein Ergebnis. Weitere Informationen zu AWS KMS Bedingungsschlüsseln in IAM-Richtlinienanweisungen finden Sie unter AWS KMS Bedingungsschlüssel.

Wenn IAM Access Analyzer einen KMS-Schlüssel analysiert, liest es Schlüsselmetadaten, wie z. B. die Schlüsselrichtlinie und die Liste der Berechtigungen. Wenn die Schlüsselrichtlinie der Rolle des IAM Access Analyzers nicht erlaubt, die Schlüssel-Metadaten zu lesen, wird eine Fehlermeldung „Zugriff verweigert“ generiert. Wenn beispielsweise die folgende beispielhafte Richtlinienanweisung die einzige Richtlinie ist, die auf einen Schlüssel angewendet wird, führt dies zu einer Fehlermeldung „Zugriff verweigert“ in IAM Access Analyzer.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
```

```
}
```

Da diese Anweisung nur der Rolle Admin aus dem AWS Konto 111122223333 den Zugriff auf den Schlüssel ermöglicht, wird die Fehlermeldung „Zugriff verweigert“ generiert, da IAM Access Analyzer den Schlüssel nicht vollständig analysieren kann. Ein Fehlerergebnis wird in der Tabelle Ergebnisse in rotem Text angezeigt. Das Ergebnis sieht wie folgt aus.

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Wenn Sie einen KMS-Schlüssel erstellen, hängen die Zugriffsberechtigungen für den Schlüssel davon ab, wie Sie den Schlüssel erstellen. Wenn Sie den Fehler „Zugriff verweigert“ für eine Schlüsselressource erhalten, wenden Sie die folgende Richtlinienanweisung auf die Schlüsselressource an, um IAM Access Analyzer die Berechtigung zum Zugriff auf den Schlüssel zu erteilen.

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Nachdem Sie das Ergebnis „Zugriff verweigert“ für eine KMS-Schlüsselressource erhalten haben und dann das Ergebnis durch Aktualisieren der Schlüsselrichtlinie auflösen, wird das Ergebnis auf

den Status „Gelöst“ aktualisiert. Bei Richtlinienanweisungen oder Schlüsselbewilligungen, die einer externen Entität die Berechtigung für den Schlüssel gewähren, werden möglicherweise zusätzliche Ergebnisse für die Schlüsselressource angezeigt.

AWS Lambda Funktionen und Schichten

Für AWS Lambda Funktionen analysiert IAM Access Analyzer Richtlinien, einschließlich Bedingungsanweisungen in einer Richtlinie, die einer externen Entität Zugriff auf die Funktion gewähren. Mit Lambda können Sie Funktionen, Versionen, Aliasnamen und Ebenen einzigartige ressourcenbasierte Richtlinien zuordnen. IAM Access Analyzer meldet externen Zugriff auf der Grundlage ressourcenbasierter Richtlinien, die Funktionen und Ebenen zugeordnet sind. IAM Access Analyzer meldet keinen externen Zugriff auf der Grundlage von ressourcenbasierten Richtlinien, die mit Aliasen verknüpft sind, und bestimmten Versionen, die mit einem qualifizierten ARN aufgerufen werden.

Weitere Informationen finden Sie unter Verwenden von ressourcenbasierten Richtlinien für Lambda und Verwenden von Versionen im AWS Lambda Entwicklerhandbuch.

Amazon Simple Queue Service Warteschlangen als Ziele

Für Amazon-SQS-Warteschlangen analysiert IAM Access Analyzer Richtlinien, einschließlich Bedingungsanweisungen in einer Richtlinie, die einer externen Entität Zugriff auf eine Warteschlange gewähren.

AWS Secrets Manager Geheimnisse

Bei AWS Secrets Manager Geheimnissen analysiert IAM Access Analyzer Richtlinien, einschließlich Bedingungsanweisungen in einer Richtlinie, die es einer externen Entität ermöglichen, auf ein Geheimnis zuzugreifen.

Amazon Simple Notification Service-Themen

IAM Access Analyzer analysiert ressourcenbasierte Richtlinien, die an Amazon SNS-Themen angefügt sind, einschließlich Bedingungsanweisungen in den Richtlinien, die externen Zugriff auf ein Thema zulassen. Mithilfe einer ressourcenbasierten Richtlinie können Sie externen Konten erlauben, Amazon-SNS-Aktionen wie das Abonnieren und Veröffentlichen von Themen durchzuführen. Ein Amazon-SNS-Thema ist von außen zugänglich, wenn Auftraggeber von einem Konto außerhalb Ihrer Vertrauenszone Vorgänge mit dem Thema durchführen können. Wenn Sie beim Erstellen eines Amazon-SNS-Themas `Everyone` in Ihrer Richtlinie auswählen, machen Sie das Thema

öffentlich zugänglich. `AddPermission` ist eine weitere Möglichkeit, einem Amazon-SNS-Thema eine ressourcenbasierte Richtlinie hinzuzufügen, die externen Zugriff zulässt.

Volume-Snapshots von Amazon Elastic Block Store

Volume-Snapshots von Amazon Elastic Block Store verfügen nicht über ressourcenbasierten Richtlinien. Ein Snapshot wird über die Amazon-EBS-Freigabeberechtigungen freigegeben. Für Amazon-EBS-Volume-Snapshots analysiert IAM Access Analyzer Zugriffssteuerungslisten, die einer externen Entität den Zugriff auf einen Snapshot ermöglichen. Ein Amazon-EBS-Volume-Snapshot kann verschlüsselt für externe Konten freigegeben werden. Ein unverschlüsselter Datei-Überblick kann für externe Konten freigegeben werden und öffentlichen Zugriff gewähren. Freigabeeinstellungen befinden sich im `CreateVolumePermissions`-Attribut des Snapshots. Wenn Kunden eine Vorschau des externen Zugriffs auf einen Amazon-EBS-Snapshot anzeigen, können sie den Verschlüsselungsschlüssel als Indikator dafür angeben, dass der Snapshot verschlüsselt ist, ähnlich wie bei der Vorschau von IAM Access Analyzer mit Secrets-Manager-Geheimnissen.

DB-Snapshots von Amazon Relational Database Service

Amazon-RDS-DB-Snapshots verfügen nicht über ressourcenbasierte Richtlinien. Ein DB-Snapshot wird über die Berechtigungen der Amazon-RDS-Datenbank freigegeben, und nur manuelle DB-Snapshots können freigegeben werden. Für Amazon-RDS-DB-Snapshots analysiert IAM Access Analyzer Zugriffskontrolllisten, die einer externen Entität den Zugriff auf einen Snapshot ermöglichen. Unverschlüsselte DB-Snapshots können öffentlich sein. Verschlüsselte DB-Snapshots können nicht öffentlich freigegeben werden, aber sie können mit bis zu 20 anderen Konten gemeinsam genutzt werden. Weitere Informationen finden Sie unter Erstellen eines DB-Snapshots. IAM Access Analyzer betrachtet die Möglichkeit, einen manuellen Datenbank-Snapshot (z. B. in einen Amazon-S3-Bucket) als vertrauenswürdigen Zugriff zu exportieren.

> **Note**
>
> IAM Access Analyzer identifiziert keinen öffentlichen oder kontoübergreifenden Zugriff, der direkt in der Datenbank selbst konfiguriert ist. IAM Access Analyzer identifiziert nur Ergebnisse für öffentlichen oder kontoübergreifenden Zugriff, der auf dem Amazon-RDS-DB-Snapshot konfiguriert ist.

Snapshots von Service-DB-Clustern von Amazon Relational Database

Snapshots des Amazon-RDS-DB-Clusters verfügen nicht über ressourcenbasierten Richtlinien. Ein Snapshot wird über die Berechtigungen des Amazon-RDS-DB-Clusters freigegeben. Für Snapshots des Amazon-RDS-DB-Clusters analysiert IAM Access Analyzer Zugriffssteuerungslisten, die einer externen Entität den Zugriff auf einen Snapshot ermöglichen. Unverschlüsselte Cluster-Snapshots können öffentlich sein. Verschlüsselte Cluster-Snapshots können nicht öffentlich freigegeben werden. Sowohl unverschlüsselte als auch verschlüsselte Cluster-Snapshots können für bis zu 20 andere Konten freigegeben werden. Weitere Informationen finden Sie unter Erstellen eines DB-Cluster-Snapshots. IAM Access Analyzer betrachtet die Möglichkeit, einen DB-Cluster-Snapshot (z. B. in einen Amazon-S3-Bucket) als vertrauenswürdigen Zugriff zu exportieren.

> **Note**
>
> Die Ergebnisse von IAM Access Analyzer beinhalten nicht die Überwachung von Teilen von Amazon RDS-DB-Clustern und -Clones mit anderen AWS-Konto oder Organisationen, die diese verwenden. AWS Resource Access Manager IAM Access Analyzer identifiziert nur Ergebnisse für den öffentlichen oder kontoübergreifenden Zugriff, der auf dem Snapshot des Amazon-RDS-DB-Clusters konfiguriert ist.

Repositories der Amazon Elastic Container Registry

Für Amazon-ECR-Repositorys analysiert IAM Access Analyzer ressourcenbasierte Richtlinien, einschließlich Bedingungsanweisungen in einer Richtlinie, die einer externen Entität den Zugriff auf ein Repository ermöglichen (ähnlich wie andere Ressourcentypen wie Amazon-SNS-Themen und Amazon-EFS-Dateisysteme). Für Amazon-ECR-Repositorys muss ein Prinzipal über die Berechtigung zu `ecr:GetAuthorizationToken` durch eine identitätsbasierte Richtlinie verfügen, um als extern verfügbar zu gelten.

Dateisysteme des Amazon Elastic File System

Für Amazon-EFS-Dateisysteme analysiert IAM Access Analyzer Richtlinien, einschließlich Bedingungsanweisungen in einer Richtlinie, die einer externen Entität den Zugriff auf ein Dateisystem ermöglichen. Ein Amazon-EFS-Dateisystem ist extern zugänglich, wenn Prinzipale von einem Konto außerhalb Ihrer Vertrauenszone Vorgänge auf diesem Dateisystem ausführen können. Der Zugriff wird durch eine Dateisystemrichtlinie definiert, die IAM verwendet, und durch die Art und Weise, wie das Dateisystem gemountet wird. Beispielsweise gilt das Mounten Ihres Amazon-EFS-

Dateisystems in einem anderen Konto als extern zugänglich, es sei denn, dieses Konto befindet sich in Ihrer Organisation und Sie haben die Organisation als Ihre Vertrauenszone definiert. Wenn Sie das Dateisystem aus einer Virtual Private Cloud mit einem öffentlichen Subnetz mounten, ist das Dateisystem extern zugänglich. Wenn Sie Amazon EFS mit verwenden AWS Transfer Family, werden Dateisystemzugriffsanforderungen blockiert, die von einem Transfer Family Family-Server empfangen werden, der einem anderen Konto als dem Dateisystem gehört, wenn das Dateisystem öffentlichen Zugriff zulässt.

Amazon DynamoDB DynamoDB-Streams

IAM Access Analyzer generiert einen Befund, wenn eine DynamoDB-Richtlinie mindestens eine kontoübergreifende Aktion zulässt, die es einer externen Entität ermöglicht, auf einen DynamoDB-Stream zuzugreifen. Weitere Informationen zu den unterstützten kontoübergreifenden Aktionen für DynamoDB finden Sie unter IAM-Aktionen, die durch ressourcenbasierte Richtlinien unterstützt werden im Amazon DynamoDB DynamoDB-Entwicklerhandbuch.

Amazon-DynamoDB-Tabellen

IAM Access Analyzer generiert einen Befund für eine DynamoDB-Tabelle, wenn eine DynamoDB-Richtlinie mindestens eine kontoübergreifende Aktion zulässt, die es einer externen Entität ermöglicht, auf eine DynamoDB-Tabelle oder einen DynamoDB-Index zuzugreifen. Weitere Informationen zu den unterstützten kontoübergreifenden Aktionen für DynamoDB finden Sie unter IAM-Aktionen, die durch ressourcenbasierte Richtlinien unterstützt werden im Amazon DynamoDB DynamoDB-Entwicklerhandbuch.

Einstellungen für IAM Access Analyzer

Wenn Sie die Konfiguration AWS Identity and Access Management Access Analyzer in Ihrem AWS Organizations Verwaltungskonto vornehmen, können Sie ein Mitgliedskonto in der Organisation als delegierten Administrator hinzufügen, um IAM Access Analyzer für Ihre Organisation zu verwalten. Der delegierte Administrator verfügt über Berechtigungen zum Erstellen und Verwalten von Analysatoren innerhalb der Organisation. Nur das Hauptkonto kann einen delegierten Administrator hinzufügen.

Delegierter Administrator für IAM Access Analyzer

Der delegierte Administrator für IAM Access Analyzer ist ein Mitgliedskonto innerhalb der Organisation, das über Berechtigungen zum Erstellen und Verwalten von Analysatoren verfügt,

die Zugriff innerhalb der Organisation analysieren. Nur das Hauptkonto kann einen delegierten Administrator hinzufügen, entfernen oder ändern.

Wenn Sie einen delegierten Administrator hinzufügen, können Sie später zu einem anderen Konto für den delegierten Administrator wechseln. Dann verliert das frühere delegierte Administratorkonto die Berechtigung für alle Analysatoren, die mit diesem Konto zum Analysieren des Zugriffs innerhalb der Organisation erstellt wurden. Diese Analysatoren wechseln in einen deaktivierten Zustand und generieren keine neuen Ergebnisse oder aktualisieren keine vorhandenen Ergebnisse mehr. Die vorhandenen Erkenntnisse für diese Analysatoren sind ebenfalls nicht mehr zugänglich. Sie können in Zukunft wieder darauf zugreifen, indem Sie das Konto als delegierten Administrator konfigurieren. Wenn Sie wissen, dass Sie nicht dasselbe Konto als ein delegierter Administrator verwenden, sollten Sie die Analysatoren löschen, bevor Sie den delegierten Administrator ändern. Dadurch werden alle generierten Ergebnisse gelöscht. Wenn der neue delegierte Administrator neue Analysatoren erstellt, werden neue Instances derselben Ergebnisse generiert. Sie verlieren keine Ergebnisse, sie werden nur für den neuen Analysator in einem anderen Konto generiert. Außerdem können Sie mit dem Hauptkonto der Organisation, das auch über Administratorberechtigungen verfügt, weiterhin auf Ergebnisse für die Organisation zugreifen. Der neue delegierte Administrator muss neue Analysatoren erstellen, damit IAM Access Analyzer die Ressourcen in Ihrer Organisation überwachen kann.

Wenn der delegierte Administrator die AWS Organisation verlässt, werden die delegierten Administratorrechte aus dem Konto entfernt. Alle Analysatoren im Konto mit der Organisation als Vertrauenszone werden in einen deaktivierten Status verschoben. Die vorhandenen Erkenntnisse für diese Analysatoren sind ebenfalls nicht mehr zugänglich.

Wenn Sie Analysatoren erstmals im Hauptkonto konfigurieren, können Sie die Option Delegierten Administrator hinzufügen auswählen, die auf der Seite mit den Analysator-Einstellungen in der IAM-Access-Analyzer-Konsole angezeigt wird.

> **Note**
>
> IAM Access Analyzer erhebt Gebühren für die Analysatoren für ungenutzten Zugriff. Sie entsprechen der Anzahl der pro Analysator und pro Monat analysierten IAM-Rollen und -Benutzer. Wenn Sie einen Analysator für ungenutzten Zugriff im Verwaltungskonto und im delegierten Administratorkonto erstellen, werden Ihnen beide Analysatoren für ungenutzten Zugriff in Rechnung gestellt. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

So fügen Sie einen delegierten Administrator mithilfe der Konsole hinzu

1. Melden Sie sich mit dem AWS Verwaltungskonto Ihrer Organisation bei der Konsole an.
2. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
3. Wählen Sie unter Access Analyzer die Option für Analysator-Einstellungen.
4. Wählen Sie Add delegated administrator (Delegierten Administrator hinzufügen).
5. Geben Sie im Feld Delegierter Administrator die Nummer des AWS-Konto s eines Organisationsmitgliedskontos ein, um den delegierten Administrator zu erstellen.

   Das Konto muss ein Mitglied Ihrer Organisation sein.
6. Wählen Sie Änderungen speichern aus.

Um einen delegierten Administrator mithilfe der AWS CLI oder der SDKs AWS hinzuzufügen

Wenn Sie einen Analyzer erstellen, um den Zugriff auf das gesamte Unternehmen in einem delegierten Administratorkonto mithilfe der AWS CLI, AWS API (mithilfe der AWS SDKs) oder zu analysieren, müssen Sie AWS Organizations APIs verwenden AWS CloudFormation, um den Dienstzugriff für IAM Access Analyzer zu aktivieren und das Mitgliedskonto als delegierten Administrator zu registrieren.

1. Aktivieren Sie den vertrauenswürdigen Dienstzugriff für IAM Access Analyzer in. AWS Organizations Weitere Informationen finden Sie im AWS Organizations Benutzerhandbuch unter So aktivieren oder deaktivieren Sie Trusted Access.
2. Registrieren Sie mithilfe der AWS Organizations `RegisterDelegatedAdministrator`API-Operation oder des `register-delegated-administrator` AWS CLI Befehls ein gültiges Mitgliedskonto Ihrer AWS Organisation als delegierter Administrator.

Nachdem Sie den delegierten Administrator geändert haben, muss der neue Administrator Analysatoren erstellen, um den Zugriff auf die Ressourcen in Ihrer Organisation zu überwachen.

Löschen von Analysatoren

Sie können vorhandene Analysatoren für externen und ungenutzten Zugriff auf der Seite mit den Analysator-Einstellungen löschen. Wenn Sie einen Analysator löschen, werden die im Analysator angegebenen Ressourcen nicht mehr überwacht und es werden keine neuen Ergebnisse generiert. Alle Ergebnisse, die vom Analysator generiert wurden, werden gelöscht.

Bei Ergebnissen, die gelöscht wurden, weil der Analyzer, der sie generiert hat, gelöscht wurde, wird das Ereignis innerhalb EventBridge der nächsten zwei Tage nach dem Löschen des Analyzers gesendet. Nach dem Löschen des Analysators kann es bis zu 90 Tage dauern, bis die Security-Hub-Ergebnisse gelöscht sind.

So löschen Sie einen Analysator

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie unter Access Analyzer die Option für Analysator-Einstellungen.
3. Wählen Sie den zu löschenden Analysator und dann Löschen aus.
4. Geben Sie in das Textfeld **`delete`** ein und wählen Sie dann Löschen.

Archivregeln

Mit Archivregeln werden neue Ergebnisse, die den beim Erstellen der Regel definierten Kriterien entsprechen, automatisch archiviert. Sie können Archivregeln auch nachträglich anwenden, um vorhandene Ergebnisse zu archivieren, die die Archivregelkriterien erfüllen. Sie können beispielsweise eine Archivregel erstellen, um alle Ergebnisse für einen bestimmten Amazon-S3-Bucket, auf den Sie regelmäßig Zugriff gewähren, automatisch zu archivieren. Oder wenn Sie einem bestimmten Auftraggeber Zugriff auf mehrere Ressourcen gewähren, können Sie eine Regel erstellen, mit der alle neuen für den Zugriff auf diesen Auftraggeber generierten Ergebnisse automatisch archiviert werden. Auf diese Weise können Sie sich nur auf aktive Ergebnisse konzentrieren, die auf ein Sicherheitsrisiko hinweisen könnten.

Wenn Sie eine Archivregel erstellen, werden automatisch nur neue Ergebnisse archiviert, die den Regelkriterien entsprechen. Vorhandene Ergebnisse werden nicht automatisch archiviert. Wenn Sie eine Regel erstellen, können Sie bis zu 20 Werte pro Kriterium in die Regel aufnehmen. Eine Liste der Filterschlüssel, mit denen Sie eine Archivregel erstellen oder aktualisieren können, finden Sie unter Filterschlüssel für IAM Access Analyzer.

> **Note**
>
> Wenn Sie eine Archivierungsregel erstellen oder bearbeiten, validiert IAM Access Analyzer nicht die Werte, die Sie in den Filter für die Regel aufnehmen. Wenn Sie beispielsweise eine Regel hinzufügen, um ein AWS-Konto, akzeptiert IAM Access Analyzer jeden Wert im Feld, selbst wenn es sich nicht um eine gültige AWS -Kontonummer handelt.

So erstellen Sie eine Archivregel

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Access Analyzer und dann Analyzer-Einstellungen aus.
3. Wählen Sie im Abschnitt Analyzer den Analyzer aus, für den Sie eine Archivierungsregel erstellen möchten.
4. Wählen Sie auf der Registerkarte Archivregeln die Option Archivregel erstellen aus.
5. Geben Sie einen Namen für die Regel ein, wenn Sie den Standardnamen ändern möchten.
6. Wählen Sie im Abschnitt Rule (Regel) unter Criteria (Kriterien) eine Eigenschaft aus, die der Regel entspricht.
7. Wählen Sie eine Bedingung für den Eigenschaftswert aus, z. B. Enthält, Ist oder Nicht gleich.

   Die verfügbaren Operatoren hängen von der gewählten Eigenschaft ab.
8. Fügen Sie optional zusätzliche Werte für die Eigenschaft hinzu oder fügen Sie zusätzliche Kriterien für die Regel hinzu. Um bei Ergebnissen mit externem Zugriff sicherzustellen, dass Ihre Regel keine neuen Ergebnisse für den öffentlichen Zugriff archiviert, können Sie auch das Kriterium Öffentlicher Zugriff einschließen und auf false setzen.

   Um einen weiteren Wert als Kriterium hinzuzufügen, wählen Sie Add another value (Weiteren Wert hinzufügen). Um ein weiteres Kriterium für die Regel hinzuzufügen, wählen Sie die Schaltfläche Kriterium hinzufügen.
9. Wenn Sie mit dem Hinzufügen von Kriterien und Werten fertig sind, wählen Sie Create rule (Regel erstellen), um die Regel nur auf neue Ergebnisse anzuwenden. Wählen Sie Create and archive active findings (Aktive Ergebnisse erstellen und archivieren), um neue und vorhandene Ergebnisse basierend auf den Regelkriterien zu archivieren. Im Abschnitt Results (Ergebnisse) können Sie die Liste der aktiven Ergebnisse überprüfen, für die die Archivregel gilt.

Um beispielsweise eine Regel für Ergebnisse bei externem Zugriff zu erstellen, die automatisch alle Ergebnisse für Amazon-S3-Buckets archiviert: Wählen Sie den Ressourcentyp und dann Ist für den Operator aus. Wählen Sie als Nächstes das S3-Bucket aus der Liste Werte aus.

Um eine Regel für ungenutzte Zugriffsergebnisse zu erstellen, durch die automatisch alle Ergebnisse für ein bestimmtes Konto archiviert werden, wählen Sie Konto des Ressourceneigentümers und dann Ist Gleich als Bedingung aus. Geben Sie die AWS-Konto ID in das Textfeld Wert ein.

Definieren Sie weiterhin Kriterien, um die Regel auf Ihre Umgebung anzupassen, und wählen Sie dann Regel erstellen aus.

Wenn Sie eine neue Regel erstellen und mehrere Kriterien hinzufügen, können Sie ein einzelnes Kriterium aus der Regel entfernen, indem Sie Remove this criterion (Dieses Kriterium entfernen) auswählen. Sie können einen für ein Kriterium hinzugefügten Wert entfernen, indem Sie Remove value (Wert entfernen) auswählen.

So bearbeiten Sie eine Archivregel

1. Wählen Sie den Namen der zu bearbeitenden Regel in der Spalte Name aus.

   Sie können jeweils nur eine Archivregel bearbeiten.
2. Fügen Sie neue Kriterien hinzu oder entfernen Sie ein vorhandenes Kriterium bzw. die Werte für jedes Kriterium.
3. Wählen Sie Save changes (Änderungen speichern), um die Regel nur auf neue Ergebnisse anzuwenden. Wählen Sie Save and archive active findings (Aktive Ergebnisse speichern und archivieren), um neue und vorhandene Ergebnisse basierend auf den Regelkriterien zu archivieren.

So löschen Sie eine Archivregel

1. Aktivieren Sie das Kontrollkästchen für die Regel, die Sie löschen möchten.
2. Wählen Sie Löschen.
3. Geben Sie **`delete`** in das Bestätigungsdialogfeld Archivregel löschen ein und wählen Sie dann Löschen.

Die Regeln werden nur aus dem Analysator in der aktuellen Region gelöscht. Sie müssen Archivregeln für jeden Analysator, den Sie in anderen Regionen erstellt haben, separat löschen.

Überwachung AWS Identity and Access Management Access Analyzer mit Amazon EventBridge

Anhand der Informationen in diesem Thema erfahren Sie, wie Sie die Ergebnisse von IAM Access Analyzer überwachen und auf Vorschauen mit Amazon zugreifen können. EventBridge EventBridge ist die neue Version von Amazon CloudWatch Events.

Ergebnis-Ereignisse

IAM Access Analyzer sendet EventBridge für jedes generierte Ergebnis ein Ereignis an, wenn der Status eines vorhandenen Ergebnisses geändert wird und wenn ein Ergebnis gelöscht wird. Um Ergebnisse und Benachrichtigungen über Ergebnisse zu erhalten, müssen Sie in Amazon eine Ereignisregel erstellen EventBridge. Bei Erstellung einer Ereignisregel können Sie auch eine Zielaktion angeben, die basierend auf der Regel ausgelöst werden soll. Sie könnten beispielsweise eine Ereignisregel erstellen, die ein Amazon-SNS-Thema auslöst, wenn ein Ereignis für ein neues Ergebnis von IAM Access Analyzer empfangen wird.

Zugriff auf Ereignisse in der Vorschau

IAM Access Analyzer sendet EventBridge für jede Zugriffsvorschau und Statusänderung ein Ereignis an. Dazu gehört ein Ereignis, wenn die Zugriffsvorschau zum ersten Mal erstellt wird (Status Erstellen), wenn die Zugriffsvorschau abgeschlossen ist (Status Abgeschlossen), oder wenn die Erstellung der Zugriffsvorschau fehlgeschlagen ist (Status Fehlgeschlagen). Um Benachrichtigungen über Zugriffsvorschauen zu erhalten, müssen Sie in eine Ereignisregel erstellen. EventBridge Bei Erstellung einer Ereignisregel können Sie auch eine Zielaktion angeben, die basierend auf der Regel ausgelöst werden soll. Sie könnten beispielsweise eine Ereignisregel erstellen, die ein Amazon-SNS-Thema auslöst, wenn ein Ereignis für eine abgeschlossene Zugriffsvorschau von IAM Access Analyzer empfangen wird.

Häufigkeit der Ereignisbenachrichtigung

IAM Access Analyzer sendet Ereignisse für neue Ergebnisse und Ergebnisse mit Statusaktualisierungen EventBridge innerhalb von etwa einer Stunde, nachdem das Ereignis in Ihrem Konto eingetreten ist. IAM Access Analyzer sendet Ereignisse auch EventBridge dann, wenn ein behobenes Ergebnis gelöscht wurde, weil die Aufbewahrungsfrist abgelaufen ist. Bei Ergebnissen, die gelöscht werden, weil der Analyzer, der sie generiert hat, gelöscht wurde, wird das Ereignis EventBridge etwa 24 Stunden nach dem Löschen des Analyzers gesendet. Bei Löschung eines Ergebnisses wird der Status des Ergebnisses nicht geändert. Stattdessen wird das `isDeleted`-Attribut auf „`true`“ festgelegt. IAM Access Analyzer sendet auch Ereignisse für neu erstellte Zugriffsvorschauen und Änderungen des Zugriffsvorschaustatus an. EventBridge

Beispiele für Ereignisse mit Ergebnissen für externen Zugriff

Im Folgenden finden Sie ein Beispiel für ein IAM Access Analyzer-Ereignis zur Suche nach einem externen Zugriff, an das gesendet wurde. EventBridge In der `id` Liste ist die ID für das Ereignis in

EventBridge aufgeführt. Weitere Informationen finden Sie unter Ereignisse und Ereignismuster unter EventBridge.

Im `detail`-Objekt beziehen sich die Werte für die Attribute „`accountId`“ und „`region`“ auf das Konto und die Region, das bzw. die bei der Suche gemeldet wurden. Das `isDeleted`-Attribut gibt an, ob das Ereignis aus dem zu löschenden Ergebnis stammt. Die `id` ist die Such-ID. Das `resources`-Array ist ein Singleton mit dem ARN des Analyzers, der das Ergebnis erzeugt hat.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::my-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

IAM Access Analyzer sendet auch Ereignisse an, um Fehler EventBridge zu finden. Ein Fehlerergebnis ist ein Ergebnis, das erzeugt wird, wenn IAM Access Analyzer die Ressource nicht analysieren kann. Ereignisse für Fehlerergebnisse umfassen ein `error`-Attribut, wie im folgenden Beispiel dargestellt.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::my-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

Beispiel für Ereignisse im Zusammenhang mit Ergebnissen für ungenutzten Zugriff

Im Folgenden finden Sie ein Beispiel für ein IAM Access Analyzer-Ereignis zur Suche nach ungenutztem Zugriff, an das gesendet wurde. EventBridge In der `id` Liste ist die ID für das Ereignis in EventBridge aufgeführt. Weitere Informationen finden Sie unter Ereignisse und Ereignismuster unter EventBridge.

Im `detail`-Objekt beziehen sich die Werte für die Attribute „`accountId`“ und „`region`“ auf das Konto und die Region, das bzw. die bei der Suche gemeldet wurden. Das `isDeleted`-Attribut gibt an, ob das Ereignis aus dem zu löschenden Ergebnis stammt. Die `id` ist die Such-ID.

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
        }
    }
```

IAM Access Analyzer sendet auch Ereignisse an, um Fehler EventBridge zu finden. Ein Fehlerergebnis ist ein Ergebnis, das erzeugt wird, wenn IAM Access Analyzer die Ressource nicht analysieren kann. Ereignisse für Fehlerergebnisse umfassen ein `error`-Attribut, wie im folgenden Beispiel dargestellt.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
```

```
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

Beispiel für Vorschau-Ereignisse

Das folgende Beispiel zeigt Daten für das erste Ereignis, an das gesendet wird, EventBridge wenn Sie eine Zugriffsvorschau erstellen. Das `resources`-Array ist ein Singleton mit der ARN des Analysators, mit dem die Zugriffsvorschau verknüpft ist. Im `detail`-Objekt bezieht sich `id` auf die ID der Zugriffsvorschau und `configuredResources` verweist auf die Ressource, für die die Zugriffsvorschau erstellt wurde. Die `status` ist `Creating` und bezieht sich auf den Status der Zugriffsvorschau. Die `previousStatus` ist nicht angegeben, da die Zugriffsvorschau gerade erstellt wurde.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
```

```
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Das folgende Beispiel zeigt Daten für ein Ereignis, an das EventBridge für eine Zugriffsvorschau mit einer Statusänderung von `Creating` zu gesendet wird`Completed`. Im Detailobjekt bezieht sich die`id` auf die Zugriffs-Vorschau-ID. Die `status` und `previousStatus` beziehen sich auf den Status der Zugriffsvorschau, wobei der vorherige Status `Creating` und der aktuelle Status ist `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Das folgende Beispiel zeigt Daten für ein Ereignis, an das EventBridge für eine Zugriffsvorschau mit einer Statusänderung von `Creating` zu gesendet wird`Failed`. Im `detail`-Objekt bezieht sich `id` auf die ID der Zugriffsvorschau. Die `status` und `previousStatus` beziehen sich auf den Status der Zugriffsvorschau, wobei der vorherige Status `Creating` und der aktuelle Status ist `Failed`. Das Feld `statusReason` enthält den Ursachencode, der angibt, dass die Zugriffsvorschau aufgrund einer ungültigen Ressourcenkonfiguration fehlgeschlagen ist.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Erstellen einer Ereignisregel mit der Konsole

Im folgenden Verfahren wird beschreiben, wie Sie mit der Konsole eine Ereignisregel erstellen.

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Erstellen Sie mit den folgenden Werten eine EventBridge Regel, die das Auffinden von Ereignissen überwacht oder auf Vorschauereignisse zugreift:

- Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
- Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
- Wählen Sie für Event pattern (Ereignismuster) Custom patterns (JSON editor) (Benutzerdefinierte Muster (JSON-Editor)) aus und fügen Sie eines der folgenden Beispiels-Ereignismuster in das Textfeld ein:
  - Verwenden Sie das folgende Musterbeispiel, um eine Regel basierend auf einem Ereignis mit einem Ergebnis für externen oder ungenutzten Zugriff zu erstellen:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
  ]
}
```

  - Verwenden Sie das folgende Musterbeispiel, um eine Regel zu erstellen, die nur auf einem Ereignis mit ungenutzten Zugriffsergebnissen basiert:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

> **Note**
>
> Sie können keine Regel erstellen, die nur auf einem Ereignis mit externen Zugriffsergebnissen basiert.

  - Verwenden Sie das folgende Musterbeispiel, um eine Regel basierend auf einem Zugriffsvorschau-Ereignis zu erstellen:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- Wählen Sie für Zieltypen die Option AWS Service und für Ziel auswählen ein Ziel aus, z. B. ein Amazon SNS-Thema oder eine Amazon AWS Lambda SNS-Funktion. Das Ziel wird ausgelöst, wenn ein Ereignis empfangen wird, das dem in der Regel definierten Ereignismuster entspricht.

Weitere Informationen zum Erstellen von Regeln finden Sie im EventBridge Amazon-Benutzerhandbuch unter Erstellen von EventBridge Amazon-Regeln, die auf Ereignisse reagieren.

Erstellen einer Ereignisregel mithilfe der CLI

1. Gehen Sie wie folgt vor, um eine Regel für Amazon EventBridge mithilfe von zu erstellen AWS CLI. Ersetzen Sie den Regelnamen *TestRule*durch den Namen Ihrer Regel.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"]}"
```

2. Sie können die Regel so anpassen, dass Zielaktionen nur für eine Teilmenge generierter Ergebnisse ausgelöst werden, z. B. Ergebnisse mit bestimmten Attributen. Im folgenden Beispiel wird veranschaulicht, wie eine Regel erstellt wird, die nur für Ergebnisse mit dem Status „Aktiv“ eine Zielaktion auslöst.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":[\"ACTIVE\"]}}"
```

Das folgende Beispiel zeigt, wie Sie eine Regel erstellen, die eine Zielaktion nur für Zugriffsvorschauen mit einem Status von `Creating` bis `Completed` auslöst.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Verwenden Sie den folgenden Beispielbefehl, um eine Lambda-Funktion als Ziel für die von Ihnen erstellte Regel zu definieren. Ersetzen Sie die Region und den Funktionsnamen im ARN entsprechend Ihrer Umgebung.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Fügen Sie die Berechtigungen hinzu, die zum Aufrufen des Regelziels erforderlich sind. Im folgenden Beispiel wird veranschaulicht, wie Berechtigungen für eine Lambda-Funktion erteilt werden, die den vorangegangenen Beispielen folgt.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Integrieren Sie Access Analyzer mit AWS Security Hub

AWS Security Hubbietet Ihnen einen umfassenden Überblick über Ihren Sicherheitsstatus AWS und hilft Ihnen dabei, Ihre Umgebung anhand von Industriestandards und Best Practices zu überprüfen. Security Hub sammelt Sicherheitsdaten von AWS Konten, Diensten und unterstützten Partnerprodukten von Drittanbietern und hilft Ihnen dabei, Ihre Sicherheitstrends zu analysieren und die Sicherheitsprobleme mit der höchsten Priorität zu identifizieren.

Wenn Sie in Security Hub AWS Identity and Access Management Access Analyzer integrieren, können Sie Ergebnisse von IAM Access Analyzer an Security Hub senden. Der Security Hub kann diese Erkenntnisse dann in die Analyse Ihres Sicherheitsniveaus einbeziehen.

Inhalt

So sendet IAM Access Analyzer Ergebnisse an den Security Hub

Im Security Hub werden Sicherheitsprobleme als Erkenntnisse verfolgt. Einige Ergebnisse stammen aus Problemen, die von anderen AWS Diensten oder von Drittanbietern entdeckt wurden. Security Hub verwendet ebenfalls verschiedene Regeln, um Sicherheitsprobleme zu erkennen und Ergebnisse zu generieren.

Security Hub bietet Tools zur Verwaltung von Erkenntnissen aus all diesen Quellen. Sie können Listen mit Erkenntnissen anzeigen und filtern und Details zu einer Erkenntnis anzeigen. Siehe .Ergebnisse anzeigen im AWS Security Hub -Leitfaden. Sie können auch den Status einer Untersuchung zu einer Erkenntnis nachverfolgen. Siehe Ergreifen von Maßnahmen zu Ergebnissen im AWS Security Hub -Leitfaden.

Alle Ergebnisse in Security Hub verwenden ein standardmäßiges JSON-Format, das AWS Security Finding Format (ASFF). Das ASFF enthält Details über die Ursache des Problems, die betroffenen Ressourcen und den aktuellen Status der Erkenntnis. Siehe AWS -Security Finding-Format (ASFF) im AWS Security Hub -Leitfaden.

AWS Identity and Access Management Access Analyzer ist einer der AWS Dienste, der Ergebnisse an Security Hub sendet. Bei ungenutztem Zugriff erkennt IAM Access Analyzer ungenutzten Zugriff, der IAM-Benutzern oder -Rollen gewährt wurde, und generiert für jeden von ihnen einen Befund. IAM Access Analyzer sendet diese Ergebnisse dann an Security Hub. Bei externem Zugriff erkennt IAM Access Analyzer eine Richtlinienerklärung, die den öffentlichen oder kontoübergreifenden Zugriff auf externe Prinzipale auf einer unterstützten Ressource in Ihrer Organisation oder Ihrem Konto ermöglicht. IAM Access Analyzer generiert einen Befund für den öffentlichen Zugriff, den er dann an Security Hub sendet. Für den kontenübergreifenden Zugriff sendet IAM Access Analyzer jeweils einen einzigen Befund für jeweils einen externen Prinzipal an Security Hub. Wenn IAM Access Analyzer mehrere kontenübergreifende Ergebnisse enthält, müssen Sie den Security Hub Hub-Befund für den einzelnen externen Prinzipal auflösen, bevor IAM Access Analyzer den nächsten kontenübergreifenden Befund liefert. Eine vollständige Liste der externen Prinzipale mit

kontenübergreifendem Zugriff außerhalb der Vertrauenszone für den Analyzer finden Sie in IAM Access Analyzer.

Ergebnisarten, die von IAM Access Analyzer sendet

IAM Access Analyzer sendet die Ergebnisse unter Verwendung des AWS Security Finding Format (ASFF) an den Security Hub. In ASFF gibt das `Types`-Feld die Art der Erkenntnis an. Die Ergebnisse von IAM Access Analyzer können die folgenden Werte für `Types` haben.

- Ergebnisse zum externen Zugriff – Auswirkungen/Offenlegung von Daten/gewährter externer Zugriff
- Ergebnisse des externen Zugriffs — Software- und Konfigurationsprüfungen/Bewährte Methoden zur AWS Sicherheit/Externer Zugriff gewährt
- Ergebnisse bei ungenutztem Zugriff — Software- und AWS Konfigurationsprüfungen/Bewährte Sicherheitsvorgehen/Ungenutzte Berechtigungen
- Ergebnisse ungenutzter Zugriffe — Software- und Konfigurationsprüfungen/ Bewährte Sicherheitsvorgehen/Ungenutzte IAM-Rolle AWS
- Ergebnisse ungenutzter Zugriffe — Software- und Konfigurationsprüfungen/ Bewährte Sicherheitsvorgehen/Unbenutztes IAM-Benutzerkennwort AWS
- Ergebnisse ungenutzter Zugriffe — Software- und Konfigurationsprüfungen/ Bewährte Methoden zur Sicherheit/Unbenutzter AWS IAM-Benutzerzugriffsschlüssel

Latenz für das Senden von Erkenntnissen

Wenn IAM Access Analyzer ein neues Ergebnis erstellt, wird es normalerweise innerhalb von 30 Minuten an den Security Hub gesendet. In seltenen Fällen und unter bestimmten Bedingungen wird IAM Access Analyzer nicht mitgeteilt, dass eine Richtlinie hinzugefügt oder aktualisiert wurde. So kann beispielsweise eine Änderung der Einstellungen für die Sperrung des öffentlichen Zugriffs auf Amazon S3-Kontoebene bis zu 12 Stunden dauern. Wenn bei der AWS CloudTrail Protokollzustellung ein Zustellungsproblem auftritt, löst die Änderung der Richtlinie außerdem keinen erneuten Scan der Ressource aus, die im Ergebnis gemeldet wurde. In diesem Fall analysiert IAM Access Analyzer die neue oder aktualisierte Richtlinie bei der nächsten periodischen Überprüfung stattfindet.

Wiederholen, wenn der Security Hub nicht verfügbar ist

Wenn der Security Hub nicht verfügbar ist, versucht IAM Access Analyzer in regelmäßigen Abständen, die Ergebnisse zu senden.

Vorhandene Ergebnisse im Security Hub aktualisieren

Nachdem es ein Ergebnis an Security Hub gesendet hat, AWS Identity and Access Management Access Analyzer sendet es Updates, um zusätzliche Beobachtungen der Findungsaktivität widerzuspiegeln, an Security Hub. Die Aktualisierungen werden im selben Ergebnis abgebildet.

Da IAM Access Analyzer Ergebnisse zum externen Zugriff pro Ressource gruppiert, ist das Ergebnis für eine Ressource im Security Hub aktiv, wenn mindestens eines der Ergebnisse für die Ressource in IAM Access Analyzer aktiv ist. Wenn alle Ergebnisse in IAM Access Analyzer für eine Ressource archiviert oder gelöst werden, wird das Security-Hub-Ergebnis archiviert. Das Security-Hub-Ergebnis wird aktualisiert, wenn Sie den Richtlinienzugriff zwischen öffentlichem und kontoübergreifendem Zugriff ändern. Diese Aktualisierung kann Änderungen am Typ, Titel, an der Beschreibung und am Schweregrad des Ergebnisses umfassen.

IAM Access Analyzer gruppiert Ergebnisse zum ungenutzten Zugriff nicht nach Ressourcen. Wenn also ein Ergebnis zum ungenutzten Zugriff in IAM Access Analyzer gelöst wird, wird das Security-Hub-Ergebnis gelöst. Das Security-Hub-Ergebnis wird aktualisiert, wenn Sie den IAM-Benutzer oder die IAM-Rolle aktualisieren, die das Ergebnis zum ungenutzten Zugriff generiert hat.

Anzeigen von IAM-Access-Analyzer-Ergebnissen im Security Hub

Um Ihre IAM-Access-Analyzer-Ergebnisse in Security Hub anzuzeigen, wählen Sie auf der Übersichtsseite im Abschnitt AWS: IAM Access Analyzer die Option Ergebnisse anzeigen. Alternativ können Sie im Navigationsbereich die Option „Ergebnisse“ auswählen. Anschließend können Sie die Ergebnisse filtern, sodass nur AWS Identity and Access Management Access Analyzer Ergebnisse angezeigt werden, indem Sie das Feld Produktname: mit dem Wert von auswählen**`IAM Access Analyzer`**.

Interpretieren von Ergebnissen von IAM Access Analyzer von Namen im Security Hub

AWS Identity and Access Management Access Analyzer sendet die Ergebnisse mithilfe des AWS Security Finding Formats (ASFF) an Security Hub. In ASFF gibt das Feld „Typen“ den Ergebnistyp. ASFF-Typen verwenden ein anderes Benennungsschema als. AWS Identity and Access Management Access Analyzer Die folgende Tabelle enthält Details zu allen ASFF-Typen, die mit AWS Identity and Access Management Access Analyzer Ergebnissen verknüpft sind, so wie sie in Security Hub erscheinen.

ASFF-Ergebnistyp	Security Hub	Beschreibung
Auswirkungen/Offenlegung von Daten/Externer Zugriff gewährt	<resource ARN> ermöglicht öffentlichen Zugang	Eine ressourcenbasierte Richtlinie, die der Ressource zugewiesen ist, ermöglich t allen externen Prinzipalen den öffentlichen Zugriff auf die Ressource.
Software- und Konfigura tionsprüfungen/Bewährte AWS Sicherheitsvorgehen/Externer Zugriff gewährt	<resource ARN> ermöglicht kontoübergreifenden Zugriff	Eine ressourcenbasierte Richtlinie, die der Ressource zugewiesen ist, ermöglicht dem Analysator kontoüber greifenden Zugriff auf externe Prinzipale außerhalb der Vertrauenszone.
Software- und AWS Konfigura tionsprüfungen/ Bewährte Methoden zur Sicherheit/ Ungenutzte Berechtigungen	<resource ARN> enthält ungenutzte Berechtigungen	Ein Benutzer oder eine Rolle enthält ungenutzte Service- und Aktionsberechtigungen.
Software- und Konfigura tionsprüfungen/ Bewährte Methoden zur Sicherheit/ Ungenutzte IAM-Rolle AWS	<resource ARN> enthält ungenutzte IAM-Rolle	Ein Benutzer oder eine Rolle enthält eine ungenutzte IAM-Rolle.
Software- und Konfigura tionsprüfungen/ Bewährte Methoden zur Sicherheit/ Unbenutztes IAM-Benut zerkennwort AWS	<resource ARN> enthält ungenutztes IAM-Benut zerpasswort	Ein Benutzer oder eine Rolle enthält ein ungenutztes IAM-Benutzerpasswort.
Software- und Konfigura tionsprüfungen/ Bewährte Methoden zur Sicherhei	<resource ARN> enthält einen ungenutzten IAM-Benutzer-Zugriffsschlüssel	Ein Benutzer oder eine Rolle enthält einen ungenutzten IAM-Benutzer-Zugriffsschlüs sel.

ASFF-Ergebnistyp	Security Hub	Beschreibung
t/Unbenutzter IAM-Benut zerzugriffsschlüssel AWS		

Typische Ergebnisse von IAM Access Analyzer

IAM Access Analyzer sendet Ergebnisse unter Verwendung des AWS -Security Finding-Formats (ASFF) an den Security Hub.

Hier ist ein Beispiel für ein typisches Ergebnis von IAM Access Analyzer für Ergebnisse für externen Zugriff.


```
{
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::my-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::my-bucket/ allows cross-account access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::my-bucket/ allows cross-account access
 from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
```

```
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::my-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
}
```

Hier ist ein Beispiel für ein typisches Ergebnis von IAM Access Analyzer für Ergebnisse für ungenutzten Zugriff.

```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
```

```
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
```

```
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```

Aktivieren und Konfigurieren der Integration

Um die Integration mit Security Hub verwenden zu können, müssen Sie den Security Hub aktivieren. Informationen zur Aktivierung von Security Hub finden Sie unter Einrichten von Security Hub im AWS Security Hub -Leitfaden.

Wenn Sie sowohl als auch IAM Access Analyzer Security Hub, wird die Integration automatisch aktiviert. IAM Access Analyzer beginnt sofort, Erkenntnisse an den Security Hub zu senden.

So beenden Sie das Senden von Ergebnissen

Um keine Ergebnisse mehr an Security Hub zu senden, können Sie entweder die Security Hub-Konsole oder die API verwenden.

Siehe Deaktivieren und Aktivieren des Flows von Ergebnissen aus einer Integration (Konsole) oder Deaktivieren des Flows von Ergebnissen aus einer Integration (Security Hub-API, AWS CLI) im AWS Security Hub -Leitfaden.

Protokollieren von IAM Access Analyzer-API-Aufrufen mit AWS CloudTrail

IAM Access Analyzer ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Dienst in IAM Access Analyzer ausgeführt wurden. CloudTrail erfasst alle API-Aufrufe für IAM Access Analyzer als Ereignisse. Die erfassten Aufrufe umfassen Aufrufe von der IAM-Access-Analyzer-Konsole und Codeaufrufe an die API-Operationen des IAM Access Analyzer.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für IAM Access Analyzer. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen.

Anhand der von CloudTrail gesammelten Informationen können Sie die Anfrage an IAM Access Analyzer, 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 CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

Informationen zu IAM Access Analyzer finden Sie unter CloudTrail

CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn in IAM Access Analyzer eine Aktivität auftritt, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen im CloudTrail Ereignisverlauf in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS Konto, einschließlich Ereignissen für IAM Access Analyzer, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole erstellen, gilt der Trail standardmäßig für alle AWS 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 IAM Access Analyzer-Aktionen werden von der IAM Access Analyzer API-Referenz protokolliert CloudTrail und sind in dieser Dokumentation dokumentiert. Beispielsweise generieren Aufrufe von `CreateArchiveRule` und `ListFindings` Aktionen Einträge in den CloudTrail Protokolldateien. `CreateAnalyzer`

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 Root- oder AWS Identity and Access Management (IAM-) Benutzeranmeldedaten 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.

Grundlagen zu Protokolldateieinträgen von IAM Access Analyzer

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.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der den `CreateAnalyzer` Vorgang demonstriert, der von einer Sitzung mit übernommener Rolle mit dem Namen „`Alice-tempcreds`14. Juni 2021“ ausgeführt wurde. Die Rollensitzung wurde von der Rolle namens `admin-tempcreds` ausgestellt.

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
```

```
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Filterschlüssel für IAM Access Analyzer

Sie können die folgenden Filterschlüssel verwenden, um eine Archivierungsregel (`CreateArchiveRule`) zu definieren, eine Archivregel zu aktualisieren (`UpdateArchiveRule`), eine Ergebnisliste (`ListFindings`und `ListFindingsV2`) abzurufen oder eine Liste von Zugriffsvorschauergebnissen für eine Ressource abzurufen (`ListAccessPreviewFindings`). Es gibt keinen Unterschied zwischen der Verwendung der IAM-API und AWS CloudFormation der Konfiguration von Archivierungsregeln.

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
Ressource	Der ARN, der die Ressource eindeutig identifiziert, auf die der externe Auftraggeber Zugriff hat. Weitere Informationen finden Sie unter Amazon-Ressourcennamen (ARNs).	String	 Ja	 Ja	 Ja
ResourceTyp `AWS::IAM::Role | AWS::KMS::Key | AWS::Lambda::Function | AWS::Lambda::LayerVersion | AWS::S3::Bucket | AWS::S3Express::DirectoryBucket | AWS::SQS:`	Der Typ der Ressource, auf die der externe Auftraggeber Zugriff hat.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
:Queue | AWS::SecretsManager::Secret | AWS::EFS::FileSystem | AWS::EC2::Snapshot | AWS::ECR::Repository | AWS::RDS::DBSnapshot | AWS::RDS::DBClusterSnapshot | AWS::SNS::Topic | AWS::DynamoDB::Stream | AWS::DynamoDB::Table					

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
resourceOwnerAccount	Die 12-stellige AWS Konto-ID, der die Ressource gehört. Weitere Informationen finden Sie unter AWS - Kontenkennungen.	String	 Ja	 Ja	 Ja
isPublic	Gibt an, ob das Ergebnis eine Ressource meldet, die über eine Richtlinie verfügt, die öffentlichen Zugriff zulässt.	Boolesch	 Ja	 Ja	 Ja
FindingType `UnusedIAMRole | UnusedIAMUserAccessKey | UnusedIAMUserPassword | UnusedPermission`	Der Typ des -Ergebnisses. Sie können nur nach dem Suchtyp für ungenutzte Zugriffsergebnisse filtern.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
status `ACTIVE | ARCHIVED | RESOLVED`	Der aktuelle Status der Aufgabenausführung.	String	 No	 Ja	 Ja
error	Gibt den Fehler an, der für das Ergebnis gemeldet wurde.	String	 Ja	 Ja	 Ja
principal.AWS	Das Konto gewährte Zugriff auf die Ressource im Feld `Principal` des Ergebnisses. Geben Sie die 12-stellige AWS Konto-ID oder den ARN des externen AWS Benutzers oder der externen Rolle ein. Weitere Informationen finden Sie unter AWS - Kontenkennungen.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
principal.Federated	Der ARN der Verbundidentität, die Zugriff auf die Ressource im Ergebnis hat. Weitere Informationen finden Sie unter Identitätsanbieter und Verbund.	String	 Ja	 Ja	 Ja
condition.aws:PrincipalArn	Der ARN des Auftraggebers (IAM-Benutzer, -Rolle oder -Gruppe), der als Bedingung für den Ressourcenzugriff angegeben wird. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja
condition.aws:PrincipalOrgID	Die Organisations-ID des Auftraggebers, die als Bedingung für den Ressourcenzugriff angegeben wird. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
condition.aws:PrincipalOrgPaths	Die Organisations- oder Organisationseinheits(OU)-ID, die als Bedingung für den Ressourcenzugriff angegeben wird. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja
condition.aws:SourceIp	Die IP-Adresse, die dem Auftraggeber Zugriff auf die Ressource ermöglicht, wenn die angegebene IP-Adresse verwendet wird. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.	IP-Adresse	 Ja	 Ja	 Ja
condition.aws:SourceVpc	Die VPC-ID, die dem Auftraggeber Zugriff auf die Ressource ermöglicht, wenn die angegebene VPC verwendet wird. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
condition.aws:UserId	Die Benutzer-ID des IAM-Benutzers von einem externen Konto, die als Bedingung für den Zugriff auf die Ressource angegeben wird. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja
condition.cognito-identity.amazonaws.com:aud	Die Amazon Cognito-Identitätspool-ID, die als Bedingung für den IAM-Rollenzugriff bei der Suche angegeben wurde. Weitere Informationen finden Sie unter IAM und AWS STS Condition Context Keys.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
condition.graph.facebook.com:app_id	Die Facebook-Anwendungs-ID (oder Site-ID), die als Bedingung angegeben wird, um eine Anmeldung mit Facebook-Verbundzugriff auf die IAM-Rolle im Ergebnis zu ermöglichen. Weitere Informationen finden Sie unter IAM- und AWS STS Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja
condition.accounts.google.com:aud	Die Google-Anwendungs-ID, die als Bedingung für den Zugriff auf die IAM-Rolle angegeben wurde. Weitere Informationen finden Sie unter IAM- und AWS STS Bedingungskontextschlüssel.	String	 Ja	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
condition.kms:CallerAccount	Die AWS Konto-ID, der die aufrufende Entität (IAM-Benutzer, Rolle oder Kontostammbenutzer) gehört, die von Diensten verwendet wird. AWS KMS Weitere Informationen finden Sie unter Bedingungsschlüssel für AWS Key Management Service.	String	 Ja	 Ja	 Ja
condition.www.amazon.com:app_id	Die Amazon-Anwendungs-ID (oder Site-ID), die als Bedingung angegeben wird, um eine Login with Amazon-Verbundzugriff auf die Rolle zu ermöglichen. Weitere Informationen hierzu finden Sie unter	String	 Ja	 Ja	 Ja
id	Die ID des Ergebnisses.	String	 No	 Ja	 Ja

Kriterium	Beschreibung	Typ	Archivierungsregel	Liste der Ergebnisse	Auflistung der Ergebnisse der Zugriffsvorschau
ChangeType (Typ der Änderung)	Stellt Kontext dazu bereit, wie das Ergebnis der Zugriffsvorschau im Vergleich zu dem in IAM Access Analyzer identifizierten vorhanden en Zugriff abschneidet.	String	 Nein	 Nein	 Ja
existingFindingId	Die vorhandene ID des Ergebnisses in IAM Access Analyzer, die nur für vorhandene Ergebniss e in der Zugriffsvorschau bereitgestellt wird.	String	 Nein	 Nein	 Ja
existingFindingStatus	Der vorhandene Status der Suche, der nur für vorhandene Ergebnisse in der Zugriffsvorschau bereitgestellt wird.	String	 Nein	 Nein	 Ja

Verwenden von serviceverknüpften Rollen für AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer verwendet eine dienstverknüpfte IAM-Rolle. Eine serviceverknüpfte Rolle ist ein spezieller Typ einer IAM-Rolle, die direkt mit IAM Access Analyzer verknüpft ist. Dienstbezogene Rollen sind von IAM Access Analyzer vordefiniert und enthalten alle Berechtigungen, die die Funktion benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle vereinfacht das Einrichten von IAM Access Analyzer, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. IAM Access Analyzer definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur IAM Access Analyzer die Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Informationen zu anderen Services, die serviceverknüpften Rollen unterstützen, finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Berechtigungen von serviceverknüpften Rollen für AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer verwendet die dienstverknüpfte Rolle mit dem Namen AWSServiceRoleForAccessAnalyzer— Erlaube Access Analyzer, Ressourcenmetadaten für externen Zugriff zu analysieren und Aktivitäten zu analysieren, um ungenutzten Zugriff zu identifizieren.

Die AWSServiceRoleForAccessAnalyzer dienstverknüpfte Rolle vertraut darauf, dass die folgenden Dienste die Rolle übernehmen:

- `access-analyzer.amazonaws.com`

Die Rollenberechtigungsrichtlinie `AccessAnalyzerServiceRolePolicy` ermöglicht IAM Access Analyzer, Aktionen für bestimmte Ressourcen durchzuführen.

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 Rollenberechtigungen im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für IAM Access Analyzer

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie Access Analyzer in der AWS Management Console oder der AWS API aktivieren, erstellt IAM Access Analyzer die dienstverknüpfte Rolle für Sie. Dieselbe serviceverknüpfte Rolle wird in allen Regionen verwendet, in denen Sie IAM Access Analyzer aktivieren. Sowohl Ergebnisse für externen Zugriff als auch für ungenutzte Zugriffsergebnisse verwenden dieselbe servicebezogene Rolle.

> **Note**
>
> IAM Access Analyzer ist regional. Sie müssen IAM Access Analyzer in jeder Region separat aktivieren.

Wenn Sie diese serviceverknüpfte Rolle löschen, erstellt IAM Access Analyzer die Rolle neu, sobald Sie das nächste Mal einen Analysator erstellen.

Sie können auch die IAM-Konsole verwenden, um eine dienstverknüpfte Rolle mit dem Anwendungsfall Access Analyzer zu erstellen. Erstellen Sie in der AWS CLI oder der AWS API eine dienstverknüpfte Rolle mit dem `access-analyzer.amazonaws.com` Dienstnamen. Weitere Informationen finden Sie unter Erstellen einer serviceverknüpfte Rolle im IAM-Leitfaden. Wenn Sie diese serviceverknüpfte Rolle löschen, können Sie mit demselben Verfahren die Rolle erneut erstellen.

Bearbeiten einer serviceverknüpften Rolle für IAM Access Analyzer

Mit IAM Access Analyzer können Sie die AWSServiceRoleForAccessAnalyzer dienstverknüpfte Rolle nicht 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 servicezuknüpften Rolle für IAM Access Analyzer

Wenn Sie ein Feature oder einen Service, die bzw. der eine serviceverknüpfte Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise ist keine ungenutzte Entität vorhanden, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch die Ressourcen für Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

> **Note**
>
> Wenn IAM Access Analyzer die Rolle verwendet, während Sie die Ressourcen löschen möchten, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

Um IAM Access Analyzer-Ressourcen zu löschen, die von AWSServiceRoleForAccessAnalyzer

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Abschnitt Access reports (Zugriffsberichte) unter Access analyzer (Zugriffs-Analysator) die Option Analyzers (Analysatoren).
3. Aktivieren Sie das Kontrollkästchen links oben oberhalb der Liste der Analysatoren in der Tabelle Analyzers (Analysatoren), um alle Analysatoren auszuwählen.
4. Wählen Sie Löschen aus.
5. Geben Sie **`delete`** ein, und wählen Sie dann Delete (Löschen), um zu bestätigen, dass Sie die Analysatoren löschen möchten.

So löschen Sie die serviceverknüpfte Rolle mit IAM

Verwenden Sie die IAM-Konsole, die oder die AWS API AWS CLI, um die AWSServiceRoleForAccessAnalyzer serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Leitfaden.

Unterstützte Regionen für serviceverknüpfte IAM-Access-Analyzer-Rollen

IAM Access Analyzer unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter AWS Regionen und Endpunkte.

Zugang zur Vorschau

AWS IAM Access Analyzer hilft Ihnen nicht nur dabei, Ressourcen zu identifizieren, die mit einer externen Entität gemeinsam genutzt werden, sondern zeigt Ihnen auch eine Vorschau der Ergebnisse von IAM Access Analyzer, bevor Sie Ressourcenberechtigungen bereitstellen, sodass Sie überprüfen können, ob Ihre Richtlinienänderungen nur den beabsichtigten öffentlichen und kontoübergreifenden Zugriff auf Ihre Ressource gewähren. Dies hilft Ihnen, mit dem beabsichtigten externen Zugriff auf Ihre Ressourcen zu beginnen.

Sie können eine Vorschau und Validierung des öffentlichen und kontoübergreifenden Zugriffs auf Ihre Amazon S3 Buckets in der Amazon S3-Konsole. Sie können auch IAM Access Analyzer-APIs verwenden, um eine Vorschau des öffentlichen und kontoübergreifenden Zugriffs auf Ihre Amazon S3 S3-Buckets, AWS KMS Schlüssel, IAM-Rollen, Amazon SQS SQS-Warteschlangen und Secrets Manager Manager-Geheimnisse anzuzeigen, indem Sie vorgeschlagene Berechtigungen für Ihre Ressource bereitstellen.

Themen

Anzeigen des Zugriffs in der Amazon S3 Konsole

Nachdem Sie Ihre Bucket-Richtlinie in der Amazon S3 Konsole abgeschlossen haben, haben Sie die Möglichkeit, eine Vorschau des öffentlichen und kontoübergreifenden Zugriffs auf Ihren Amazon S3-Bucket anzuzeigen. Sie können überprüfen, dass Ihre Richtlinienänderungen nur den beabsichtigten externen Zugriff zulassen, bevor Sie Änderungen speichern wählen. In diesem optionalen Schritt können Sie eine Vorschau der AWS Identity and Access Management Access Analyzer Ergebnisse für Ihren Bucket anzeigen. Sie können überprüfen, ob die Richtlinienänderung neue Ergebnisse einführt oder vorhandene Ergebnisse für den externen Zugriff auflöst. Sie können diesen Validierungsschritt überspringen und Ihre Amazon S3 Bucket-Richtlinie jederzeit speichern.

Um eine Vorschau des externen Zugriffs auf Ihren Bucket anzuzeigen, benötigen Sie einen aktiven Kontenanalysator in der Region Ihres Buckets mit dem Konto als Vertrauenszone. Sie müssen auch über die erforderlichen Berechtigungen verfügen, um IAM Access Analyzer und die Vorschau des Zugriffs zu verwenden. Weitere Informationen zur Aktivierung von IAM Access Analyzer und den erforderlichen Berechtigungen finden Sie unter Aktivieren von IAM Access Analyzer.

So zeigen Sie eine Vorschau des Zugriffs auf Ihren Amazon S3 Bucket an, wenn Sie Ihre Bucket-Richtlinie erstellen oder bearbeiten

1. Sobald Sie die Erstellung oder Bearbeitung Ihrer Bucket-Richtlinie abgeschlossen haben, stellen Sie sicher, dass es sich bei Ihrer Richtlinie um eine gültige Amazon S3 Bucket-Richtlinie handelt. Der ARN der Richtlinie muss mit dem ARN des Buckets übereinstimmen und die Elemente der Richtlinie müssen gültig sein.
2. Wählen Sie unterhalb der Richtlinie unter Vorschau des externen Zugriffs einen aktiven Account Analyzer und wählen Sie dann Vorschau. Für Ihren Bucket wird eine Vorschau der IAM-Access-Analyzer-Ergebnisse generiert. In der Vorschau wird die angezeigte Amazon S3 Bucket-Richtlinie zusammen mit den vorhandenen Bucket-Berechtigungen analysiert. Dazu gehören die Bucket- und Konto-BPA-Einstellungen, Bucket-ACL, die Amazon S3 Zugriffspunkte und regionübergreifende Zugriffspunkte, die dem Bucket zugeordnet sind, sowie deren Richtlinien und BPA-Einstellungen.
3. Wenn die Zugriffsvorschau abgeschlossen ist, wird eine Vorschau der Ergebnisse von IAM Access Analyzer angezeigt. Jede Suche meldet eine Instance eines Auftraggebers außerhalb

des Kontos mit Zugriff auf Ihren Bucket, nachdem Sie die Richtlinie gespeichert haben. Sie können den Zugriff auf Ihren Bucket überprüfen, indem Sie die einzelnen Ergebnisse überprüfen. Die Kopfzeile des Ergebnisses enthält eine Zusammenfassung des Zugriffs, und Sie können das Ergebnis erweitern, um die Details des Ergebnisses zu überprüfen. Das Suchen von Badges bietet Kontext dazu, wie das Speichern der Bucket-Richtlinie den Zugriff auf den Bucket ändern würde. Sie helfen Ihnen beispielsweise zu überprüfen, ob die Richtlinienänderung neue Erkenntnisse einführt oder vorhandene Ergebnisse für den externen Zugriff löst:

   a. Neu – zeigt einen neuen externen Zugriff an, den die Richtlinie einführen würde.
   b. Gelöst – gibt eine Suche nach vorhandenem externen Zugriff an, die von der Richtlinie entfernt werden soll.
   c. Archiviert – gibt einen neuen externen Zugriff an, der automatisch archiviert wird, basierend auf den Archivregeln für den Analysator, die festlegen, wann Ergebnisse wie beabsichtigt markiert werden sollen.
   d. Existing – zeigt eine vorhandene Suche für externen Zugriff an, die unverändert bleiben würde.
   e. Öffentlich – Wenn ein Fund für den öffentlichen Zugang zu einer Ressource bestimmt ist, wird er zusätzlich zu einem der oben genannten Abzeichen mit einem öffentlichen Abzeichen versehen.

4. Wenn Sie feststellen, dass Sie den externen Zugriff nicht einführen oder entfernen wollen, können Sie die Richtlinie überarbeiten und dann erneut die Vorschau wählen, bis Sie den gewünschten externen Zugriff erreicht haben. Wenn Sie einen Fund mit der Bezeichnung Öffentlich haben, empfehlen wir Ihnen, die Richtlinie zu überarbeiten, um den öffentlichen Zugang zu entfernen, bevor Sie Änderungen speichern wählen. Die Vorschau des Zugangs ist ein optionaler Schritt, und Sie können jederzeit die Option Änderungen speichern wählen.

Vorschau des Zugriffs mit APIs von IAM Access Analyzer

Sie können IAM Access Analyzer-APIs verwenden, um eine Vorschau des öffentlichen und kontoübergreifenden Zugriffs auf Ihre Amazon S3 S3-Buckets, AWS KMS Schlüssel, IAM-Rollen, Amazon SQS SQS-Warteschlangen und Secrets Manager Manager-Geheimnisse anzuzeigen. Sie können eine Vorschau des Zugriffs anzeigen, indem Sie vorgeschlagene Berechtigungen für eine vorhandene Ressource, die Sie besitzen, oder für eine neue Ressource bereitstellen möchten.

Um eine Vorschau des externen Zugriffs auf Ihre Ressource anzuzeigen, benötigen Sie einen aktiven Kontenanalysator für das Konto und die Region der Ressource. Sie müssen auch über die

erforderlichen Berechtigungen verfügen, um IAM Access Analyzer und die Vorschau des Zugriffs zu verwenden. Weitere Informationen zur Aktivierung von IAM Access Analyzer und den erforderlichen Berechtigungen finden Sie unter Aktivieren von IAM Access Analyzer.

Um den Zugriff auf eine Ressource in der Vorschau anzuzeigen, können Sie die Operation `CreateAccessPreview` verwenden und den ARN des Analysators sowie die Konfiguration der Zugriffskontrolle für die Ressource angeben. Der Dienst gibt die eindeutige ID für die Zugriffsvorschau zurück, mit der Sie den Status der Zugriffsvorschau mit der Operation `GetAccessPreview` verwenden. Wenn der Status `Completed` ist, können Sie die Operation `ListAccessPreviewFindings` verwenden, um die für die Zugriffsvorschau generierten Ergebnisse abzurufen. Die Operationen `GetAccessPreview` und `ListAccessPreviewFindings` werden innerhalb von etwa 24 Stunden Zugriffsvorschauen und erstellte Ergebnisse abrufen.

Jedes abgerufene Ergebnis enthält Ergebnisdetails, die den Zugriff beschreiben. Ein Vorschaustatus des Ergebnisses, der beschreibt, ob das Ergebnis `Active`, `Archived`, oder `Resolved` ist nach der Bereitstellung von Berechtigungen, und ein `changeType`. Die `changeType` bietet Kontext dazu, wie das Ergebnis der Zugriffsvorschau im Vergleich zu dem in IAM Access Analyzer identifizierten vorhandenen Zugriff abschneidet:

- Neu – das Ergebnis ist für den neu eingeführten Zugang.
- Unverändert – Bei dem Vorschauergebnis handelt es sich um ein vorhandenes Ergebnis, das unverändert bleibt.
- der Änderung – Bei dem Vorschauergebnis handelt es sich um ein vorhandenes Ergebnis mit einer Statusänderung.

Die `status` und die `changeType` helfen Ihnen zu verstehen, wie die Ressourcenkonfiguration den Zugriff auf vorhandene Ressourcen ändern würde. Wenn die `changeType` `Unchanged` oder geändert ist, enthält den Status des Befunds auch die vorhandene ID und das in IAM Access Analyzer. Ein Ergebnis `Changed` mit dem Vorschaustatus `Resolved` und dem bestehenden Status `Active` bedeutet zum Beispiel, dass das bestehende Ergebnis `Active` für die Ressource durch die vorgeschlagene Änderung der Berechtigungen zu `Resolved` wird.

Mit der Operation `ListAccessPreviews` können Sie eine Liste der Zugriffsvorschauen für den angegebenen Analyzer abrufen. Dieser Vorgang ruft Informationen zur Zugriffsvorschau ab, die innerhalb von etwa einer Stunde erstellt wurden.

Wenn die Zugriffsvorschau für eine vorhandene Ressource gilt und Sie eine Konfigurationsoption nicht spezifiziert lassen, verwendet die Zugriffsvorschau standardmäßig die vorhandene

Ressourcenkonfiguration. Wenn die Zugriffsvorschau für eine neue Ressource gilt und Sie eine Konfigurationsoption nicht angegeben lassen, verwendet die Zugriffsvorschau je nach Ressourcentyp den Standardwert. Konfigurationsbeispiele für die einzelnen Ressourcentypen finden Sie weiter unten.

Vorschau des Zugriffs auf Ihren Amazon S3 Bucket

Um eine Zugriffsvorschau für einen neuen Amazon S3 Bucket oder einen vorhandenen Amazon S3-Bucket zu erstellen, den Sie besitzen, können Sie eine Bucket-Konfiguration vorschlagen, indem Sie die Amazon S3-Bucket-Richtlinie, Bucket-ACLs, Bucket-BPA-Einstellungen und Amazon S3-Zugriffspunkte, einschließlich Zugriffspunkte für mehrere Regionen, angeben, die dem Bucket zugeordnet sind.

> **Note**
>
> Bevor Sie versuchen, eine Zugriffsvorschau für einen neuen Bucket zu erstellen, empfehlen wir Ihnen, den Amazon S3 HeadBucketS3-Vorgang aufzurufen, um zu überprüfen, ob der benannte Bucket bereits existiert. Mit dieser Operation können Sie feststellen, ob ein Bucket vorhanden ist, und ob Sie Zugriffsberechtigungen für diesen Bucket besitzen.

Bucket-Richtlinie – Wenn die Konfiguration für einen vorhandenen Amazon S3 Bucket gilt und Sie die Amazon S3-Bucket-Richtlinie nicht angeben, verwendet die Zugriffsvorschau die vorhandene Richtlinie, die dem Bucket zugeordnet ist. Wenn die Zugriffsvorschau für eine neue Ressource gilt und Sie die Amazon S3 Bucket-Richtlinie nicht angeben, wird in der Zugriffsvorschau ein Bucket ohne Richtlinie angenommen. Um das Löschen einer vorhandenen Bucket-Richtlinie vorzuschlagen, können Sie eine leere Zeichenfolge angeben. Weitere Informationen zu den unterstützten Limits von Bucketrichtlinien finden Sie unter Beispiele für Bucket-Richtlinien.

Grants für Bucket-ACL – Sie können bis zu 100 ACL-Zuschüsse pro Bucket vorschlagen. Wenn die vorgeschlagene Zuschuss-Konfiguration für einen vorhandenen Bucket gilt, verwendet die Zugriffsvorschau anstelle der vorhandenen Berechtigungen die vorgeschlagene Liste der Zuschuss-Konfiguration. Andernfalls verwendet die Zugriffsvorschau die vorhandenen Berechtigungen für den Bucket.

Bucket-Zugriffspunkte – Die Analyse unterstützt bis zu 100 Zugriffspunkte, einschließlich Zugriffspunkte für mehrere Regionen, pro Bucket, einschließlich bis zu zehn neuer Zugriffspunkte, die Sie pro Bucket vorschlagen können. Wenn die vorgeschlagene Amazon S3 Zugriffspunkt-Konfiguration für einen vorhandenen Bucket gilt, verwendet die Zugriffsvorschau anstelle der

vorhandenen Zugriffspunkte die vorgeschlagene Zugriffspunkt-Konfiguration. Um einen Zugriffspunkt ohne Richtlinie vorzuschlagen, können Sie eine leere Zeichenfolge als Zugriffspunktrichtlinie angeben. Weitere Informationen zu den Beschränkungen der Zugriffspunktrichtlinien finden Sie unter Beschränkungen und Einschränkungen für Zugriffspunkte.

Blockieren des öffentlichen Zugriffs – Wenn die vorgeschlagene Konfiguration für einen vorhandenen Amazon S3 Bucket gilt und Sie die Konfiguration nicht angeben, verwendet die Zugriffsvorschau die vorhandene Einstellung. Wenn die vorgeschlagene Konfiguration für einen neuen Bucket gilt und Sie die Bereichs-BPA-Konfiguration nicht angeben, verwendet die Zugriffsvorschau `false`. Wenn die vorgeschlagene Konfiguration für einen neuen Zugangspunkt oder einen Zugangspunkt mit mehreren Regionen gilt und Sie die BPA-Konfiguration des Zugangspunkts nicht angeben, verwendet die Zugangsvorschau `true`.

Vorschau des Zugriffs auf Ihre AWS KMS -Schlüssel

Um eine Zugriffsvorschau für einen neuen AWS KMS Schlüssel oder einen vorhandenen AWS KMS Schlüssel, den Sie besitzen, zu erstellen, können Sie eine AWS KMS Schlüsselkonfiguration vorschlagen, indem Sie die Schlüsselrichtlinie und die AWS KMS Grant-Konfiguration angeben.

AWS KMS Schlüsselrichtlinie — Wenn die Konfiguration für einen vorhandenen Schlüssel gilt und Sie die Schlüsselrichtlinie nicht angeben, verwendet die Zugriffsvorschau die bestehende Richtlinie für den Schlüssel. Wenn die Zugriffsvorschau für eine neue Ressource gilt und Sie die Schlüsselrichtlinie nicht angeben, verwendet die Zugriffsvorschau die Standardschlüsselrichtlinie. Die vorgeschlagene Schlüsselrichtlinie darf keine leere Zeichenfolge sein.

AWS KMS Zuschüsse — Die Analyse unterstützt bis zu 100 KMS-Zuschüsse pro Konfiguration*.* Wenn die vorgeschlagene Grant-Konfiguration für einen vorhandenen Schlüssel gilt, verwendet die Access Preview die vorgeschlagene Liste der Grant-Konfigurationen anstelle der vorhandenen Grants. Andernfalls verwendet die Zugriffsvorschau die vorhandenen Berechtigungen für den Schlüssel.

Vorschau des Zugriffs auf Ihre IAM-Rolle

Um eine Zugriffsvorschau für eine neue IAM-Rolle oder eine vorhandene IAM-Rolle zu erstellen, die Sie besitzen, können Sie eine IAM-Rollenkonfiguration vorschlagen, indem Sie die Vertrauensrichtlinie angeben.

Vertrauensrichtlinie für Rollen – Wenn die Konfiguration für eine neue IAM-Rolle gilt, müssen Sie die Vertrauensrichtlinie angeben. Wenn die Konfiguration für eine vorhandene IAM-Rolle gilt, die

Sie besitzen und Sie die Vertrauensrichtlinie nicht vorschlagen, verwendet die Zugriffsvorschau die vorhandene Vertrauensrichtlinie für die Rolle. Die vorgeschlagene Vertrauensrichtlinie darf keine leere Zeichenfolge sein.

Vorschau des Zugriffs auf Ihre Amazon SQS Warteschlange

Um eine Zugriffsvorschau für eine neue Amazon SQS Warteschlange oder eine vorhandene Amazon SQS Warteschlange zu erstellen, die Sie besitzen, können Sie eine Amazon SQS-Warteschlangenkonfiguration vorschlagen, indem Sie die Amazon SQS-Richtlinie für die Warteschlange angeben.

Amazon SQS Warteschlangenrichtlinie – Wenn die Konfiguration für eine vorhandene Amazon SQS Warteschlange gilt und Sie die Amazon SQS-Richtlinie nicht angeben, verwendet die Zugriffsvorschau die vorhandene Amazon SQS-Richtlinie für die Warteschlange. Wenn die Zugriffsvorschau für eine neue Ressource gilt und Sie die Richtlinie nicht angeben, wird in der Zugriffsvorschau eine Amazon SQS Warteschlange ohne Richtlinie angenommen. Um eine vorhandene Amazon SQS Warteschlangenrichtlinie zu löschen, können Sie eine leere Zeichenfolge für die Amazon SQS-Richtlinie angeben.

Vorschau auf den Zugriff auf Ihr Secrets Manager Geheimnis

Um eine Zugriffsvorschau für ein neues Secrets Manager-Geheimnis oder ein vorhandenes Secrets Manager-Geheimnis, das Sie besitzen, zu erstellen, können Sie eine Secrets Manager-Konfiguration vorschlagen, indem Sie die geheime Richtlinie und den optionalen AWS KMS Verschlüsselungsschlüssel angeben.

Geheime Richtlinie – Wenn die Konfiguration für einen vorhandenen geheimen Schlüssel gilt und Sie die geheime Richtlinie nicht angeben, verwendet die Zugriffsvorschau die vorhandene Richtlinie für den geheimen Schlüssel. Wenn die Zugriffsvorschau für eine neue Ressource gilt und Sie die Richtlinie nicht angeben, wird in der Zugriffsvorschau ein Geheimnis ohne Richtlinie angenommen. Um eine vorhandene Richtlinie zu löschen, können Sie eine leere Zeichenfolge angeben.

AWS KMS Verschlüsselungsschlüssel — Wenn die vorgeschlagene Konfiguration für ein neues Geheimnis gilt und Sie die AWS KMS Schlüssel-ID nicht angeben, verwendet die Zugriffsvorschau den Standard-KMS-Schlüssel des AWS Kontos. Wenn Sie eine leere Zeichenfolge für die AWS KMS Schlüssel-ID angeben, verwendet die Zugriffsvorschau den Standard-KMS-Schlüssel des AWS Kontos.

Prüft, ob Richtlinien validiert werden

IAM Access Analyzer bietet Richtlinienprüfungen, mit deren Hilfe Sie Ihre IAM-Richtlinien überprüfen können, bevor Sie sie an eine Entität anhängen. Dazu gehören grundlegende Richtlinienprüfungen, die im Rahmen der Richtlinienvalidierung durchgeführt werden, um Ihre Richtlinie anhand der Richtliniengrammatik und der bewährten Methoden für AWS zu validieren. Sie können die Ergebnisse der Richtlinienvalidierung anzeigen, die Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge für Ihre Richtlinie enthalten.

Sie können benutzerdefinierte Richtlinienüberprüfungen verwenden, um anhand Ihrer Sicherheitsstandards nach neuen Zugriffen zu suchen. Für jede Prüfung auf bisher nicht gewährten Zugriff wird eine Gebühr erhoben. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

Themen

Richtlinienvalidierung von IAM Access Analyzer

Sie können Ihre Richtlinien mithilfe der AWS Identity and Access Management Access Analyzer Richtlinienvalidierung validieren. Sie können eine Richtlinie mithilfe des AWS CLI AWS API- oder JSON-Richtlinieneditors in der IAM-Konsole erstellen oder bearbeiten. IAM Access Analyzer validiert Ihre Richtlinie anhand der IAM-Richtliniengrammatik und anhand von bewährten Methoden für AWS. Sie können die Ergebnisse der Richtlinienvalidierung anzeigen, die Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge für Ihre Richtlinie enthalten. Diese Ergebnisse enthalten verwertbare Empfehlungen, mit denen Sie Richtlinien erstellen können, die funktionsfähig sind und den bewährten Sicherheitsmethoden entsprechen. Informationen zum Anzeigen einer Liste der grundlegenden Richtlinienüberprüfungen, die von IAM Access Analyzer ausgeführt werden, finden Sie unter Referenz zur Richtlinienprüfung von Access Analyzer.

Validieren von Richtlinien in IAM (Konsole)

Sie können Ergebnisse anzeigen, die von der IAM-Access-Analyzer-Richtlinienvalidierung generiert werden, wenn Sie eine verwaltete Richtlinie in der IAM-Konsole erstellen oder bearbeiten. Sie können diese Ergebnisse auch für Inline-Benutzer- oder Rollenrichtlinien anzeigen. IAM Access Analyzer generiert diese Ergebnisse nicht für Inline-Gruppenrichtlinien.

So zeigen Sie Ergebnisse an, die von Richtlinienüberprüfungen für IAM-JSON-Richtlinien

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Verwenden Sie eine der folgenden Methoden, um eine Richtlinie zu erstellen oder zu bearbeiten:

   a. Um eine neue verwaltete Richtlinie zu erstellen, wechseln Sie zum Richtlinien und erstellen Sie eine neue Richtlinie. Weitere Informationen finden Sie unter Erstellen von Richtlinien mit dem JSON-Editor.

   b. Um Richtlinienprüfungen für eine vorhandene, vom Kunden verwaltete Richtlinie anzuzeigen, gehen Sie zur Seite Richtlinien, wählen Sie den Namen einer Richtlinie und dann Bearbeiten aus. Weitere Informationen finden Sie unter Bearbeiten von kundenverwalteten Richtlinien (Konsole).

   c. Um Richtlinienprüfungen für eine Inline-Richtlinie für einen Benutzer oder eine Rolle anzuzeigen, gehen Sie zur Seite Benutzer oder Rollen, wählen Sie den Namen eines Benutzers oder einer Rolle aus, wählen Sie auf der Registerkarte Berechtigungen den Namen der Richtlinie aus und wählen Sie dann Bearbeiten aus. Weitere Informationen finden Sie unter Bearbeiten von kundenverwalteten Richtlinien (Konsole).
3. Wählen Sie im Richtlinien-Editor die Option JSON-Registerkarte.
4. Wählen Sie im Bereich der Richtlinienüberprüfung unterhalb der Richtlinie eine oder mehrere der folgenden Registerkarten. Die Registerkarten geben auch die Anzahl der einzelnen Suchtypen für Ihre Richtlinie an.

   - Sicherheit — Lassen Sie sich Warnungen anzeigen, wenn Ihre Richtlinie einen Zugriff zulässt, der AWS ein Sicherheitsrisiko darstellt, weil der Zugriff zu freizügig ist.
   - Fehler – Zeigt Fehler an, wenn Ihre Richtlinie Zeilen enthält, die das Funktionieren der Richtlinie verhindern.
   - Warnungen – Zeigen Sie Warnungen an, wenn Ihre Richtlinie nicht den bewährten Methoden entspricht, die Probleme jedoch keine Sicherheitsrisiken darstellen.
   - Vorschläge – Vorschläge anzeigen, wenn AWS empfiehlt Verbesserungen, die sich nicht auf die Berechtigungen der Richtlinie auswirken.
5. Überprüfen Sie die Ergebnisdetails, die von der Richtlinienüberprüfung von IAM Access Analyzer bereitgestellt werden. Jede Feststellung gibt den Speicherort des gemeldeten Problems an. Wenn Sie mehr über die Ursache des Problems und seine Behebung erfahren möchten, klicken Sie auf den Link Weitere Informationen neben dem Fund. Sie können auch auf der Referenzseite

für Access Analyzer-Richtlinienprüfungen nach der Richtlinienprüfung suchen, die mit jedem Fund verbunden ist.

6. Optional. Wenn Sie eine bestehende Richtlinie bearbeiten, können Sie eine benutzerdefinierte Richtlinienüberprüfung durchführen, um festzustellen, ob Ihre aktualisierte Richtlinie im Gegensatz zur vorhandenen Version nun Zugriff gewährt. Wählen Sie im Bereich der Richtlinienüberprüfung unterhalb der Richtlinie die Registerkarte Nach neuem Zugriff prüfen und dann Richtlinie überprüfen. Wenn die geänderten Berechtigungen bisher nicht gewährten Zugriff gewähren, wird die Anweisung im Bereich zur Richtlinienvalidierung hervorgehoben. Wenn Sie nicht beabsichtigen, neuen Zugriff zu gewähren, aktualisieren Sie die Richtlinienanweisung und wählen Sie Richtlinie überprüfen, bis kein neuer Zugriff erkannt wird. Weitere Informationen finden Sie unter IAM Access Analyzer – Benutzerdefinierte Richtlinienüberprüfungen.

   > **Note**
   >
   > Für jede Prüfung auf bisher nicht gewährten Zugriff wird eine Gebühr erhoben. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

7. Aktualisieren Sie Ihre Richtlinie, um die Ergebnisse zu beheben.

   > **Important**
   >
   > Testen Sie neue oder bearbeitete Richtlinien gründlich, bevor Sie sie in Ihren Produktionsworkflow implementieren.

8. Wählen Sie danach Next aus. Die Richtlinienvalidierung meldet Syntaxfehler, die nicht von IAM Access Analyzer gemeldet werden.

   > **Note**
   >
   > Sie können jederzeit zwischen den Registerkarten Visuell und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder auf der Registerkarte Visuell die Option Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

9. Geben Sie für neue Richtlinien 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 In dieser Richtlinie definierte Berechtigungen, um die Berechtigungen

einzusehen, die von Ihrer Richtlinie gewährt werden. Wählen Sie dann Create policy aus, um Ihre Eingaben zu speichern.

Überprüfen Sie für bestehende Richtlinien auf der Seite Überprüfen und speichern den Punkt In dieser Richtlinie definierte Berechtigungen, um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden. Aktivieren Sie das Kontrollkästchen Diese neue Version als Standard festlegen, um die aktualisierte Version als Standardversion der Richtlinie zu speichern. Wählen Sie anschließend Kopieren aus, um die Änderungen zu speichern.

Überprüfen von Richtlinien mithilfe von IAM Access Analyzer (oder API)AWS CLIAWS

Sie können Ergebnisse, die von IAM Access Analyzer-Richtlinienvalidierungen generiert wurden, im AWS Command Line Interface (AWS CLI) enthalten.

Um Ergebnisse anzuzeigen, die durch die IAM Access Analyzer-Richtlinienvalidierung (AWS CLI oder API) generiert wurden AWS

Nutzen Sie einen der Folgenden:

- AWS CLI:aws accessanalyzer validate-policy
- AWS API: ValidatePolicy

Referenz zur Richtlinienprüfung von Access Analyzer

Sie können Ihre Richtlinien mithilfe der AWS Identity and Access Management Access Analyzer Richtlinienvalidierung validieren. Sie können eine Richtlinie mithilfe des AWS CLI AWS API- oder JSON-Richtlinieneditors in der IAM-Konsole erstellen oder bearbeiten. IAM Access Analyzer validiert Ihre Richtlinie anhand der IAM-Richtliniengrammatik und anhand von bewährten Methoden für AWS. Sie können die Ergebnisse der Richtlinienvalidierung anzeigen, die Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge für Ihre Richtlinie enthalten. Diese Ergebnisse enthalten verwertbare Empfehlungen, mit denen Sie Richtlinien erstellen können, die funktionsfähig sind und den bewährten Sicherheitsmethoden entsprechen. Die grundlegenden Richtlinienüberprüfungen, die von IAM Access Analyzer bereitgestellt werden, sind unten aufgeführt. Für die Durchführung von Richtlinienvalidierungsprüfungen fallen keine zusätzlichen Gebühren an. Weitere Informationen zum Validieren von Richtlinien mit der Richtlinienvalidierung finden Sie unter Richtlinienvalidierung von IAM Access Analyzer.

Fehler – ARN Konto nicht zulässig

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Beheben des Fehlers

Entfernen Sie die Konto-ID aus dem Ressourcen-ARN. Die Ressourcen-ARNs für einige AWS Dienste unterstützen die Angabe einer Konto-ID nicht.

Amazon S3 unterstützt beispielsweise keine Konto-ID als Namespace in Bucket-ARNs. Ein Amazon S3 S3-Bucket-Name ist weltweit eindeutig, und der Namespace wird von allen AWS Konten gemeinsam genutzt. Um alle in Amazon S3 verfügbaren Ressourcentypen zu sehen, siehe Ressourcentypen, die von Amazon S3 definiert werden, in der Service Authorization Reference.

Verwandte Begriffe

- Richtlinienressourcen
- Konto-Kennungen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – ARN Region nicht zulässig

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Beheben des Fehlers

Entfernen Sie die Region aus der Ressourcen-ARN. Die Ressourcen-ARNs für einige AWS Dienste unterstützen die Angabe einer Region nicht.

IAM ist beispielsweise ein globaler Service. Der Regionsteil eines IAM-Ressourcen-ARN wird immer leer gehalten. IAM-Ressourcen sind global, genau wie ein AWS Konto heute. Nachdem Sie sich beispielsweise als IAM-Benutzer angemeldet haben, können Sie auf AWS Dienste in jeder geografischen Region zugreifen.

- Richtlinienressourcen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – Datentyp stimmt nicht überein

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Beheben des Fehlers

Aktualisieren Sie den Text, um den unterstützten Datentyp zu verwenden.

Zum Beispiel, das `Version`-Globale Bedingungsschlüssel erfordert einen `String`-Datentyp. Wenn Sie ein Datum oder eine Ganzzahl angeben, stimmt der Datentyp nicht überein.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Fehler – Doppelte Schlüssel mit unterschiedlicher Groß- und Kleinschreibung

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Beheben des Fehlers

Überprüfen Sie die ähnlichen Bedingungsschlüssel innerhalb desselben Bedingungsblocks und verwenden Sie dieselbe Großschreibung für alle Instances.

A Bedingungsblock ist der Text innerhalb des `Condition`-Element einer Policy-Anweisung. Bei Namen von Bedingungsschlüsseln wird die Groß- und Kleinschreibung nicht beachtet. Die Groß-/Kleinschreibung der Werte von Bedingungsschlüsseln hängt vom verwendeten Bedingungsoperator ab. Weitere Hinweise zur Berücksichtigung von Groß- und Kleinschreibung in Bedingungsschlüssel finden Sie unter IAM-JSON-Richtlinienelemente: Condition.

Verwandte Begriffe

- Bedingungen
- Bedingungsblock
- Globale Bedingungsschlüssel
- AWS Schlüssel für Servicebedingungen

Fehler – Ungültige Aktion

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Beheben des Fehlers

Die angegebene Aktion ist ungültig. Dies kann passieren, wenn Sie das Servicepräfix oder den Aktionsnamen falsch eingeben. Bei einigen häufig auftretenden Problemen gibt die Richtlinienprüfung eine vorgeschlagene Aktion zurück.

Verwandte Begriffe

- Richtlinienaktionen
- AWS Dienstaktionen

AWS verwaltete Richtlinien mit diesem Fehler

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Die folgenden AWS verwalteten Richtlinien enthalten ungültige Aktionen in ihren Richtlinienerklärungen. Ungültige Aktionen wirken sich nicht auf die Berechtigungen aus, die von der Richtlinie gewährt werden. Wenn Sie eine AWS verwaltete Richtlinie als Referenz für die Erstellung Ihrer verwalteten Richtlinie verwenden, AWS empfiehlt Ihnen, ungültige Aktionen aus Ihrer Richtlinie zu entfernen.

- AmazonEMR _v2 FullAccessPolicy
- CloudWatchSyntheticsFullAccess

Fehler – Ungültiges ARN Konto

In der beinhaltet AWS Management Console das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-digit account ID."
```

Beheben des Fehlers

Aktualisieren Sie die Konto-ID im Ressourcen-ARN. Konto-IDs sind 12-stellige Ganzzahlen. Informationen zum Anzeigen Ihrer Konto-ID finden Sie unter Ihre AWS Konto-ID ermitteln.

Verwandte Begriffe

- Richtlinienressourcen
- Konto-Kennungen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – Ungültiges ARN Präfix

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Beheben des Fehlers

AWS Ressourcen-ARNs müssen das erforderliche `arn:` Präfix enthalten.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – Ungültige ARN Region

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Beheben des Fehlers

Der Ressourcentyp wird in der angegebenen Region nicht unterstützt. Eine Tabelle der in den einzelnen Regionen unterstützten AWS Dienste finden Sie in der Tabelle mit den Regionen.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- Regionsnamen und -codes

Fehler – Ungültige ARN Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Beheben des Fehlers

Der Ressourcen-ARN muss mit den Spezifikationen für bekannte Ressourcentypen übereinstimmen. Informationen zum erwarteten ARN-Format für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service aus, um seine Ressourcentypen und ARN Formate anzuzeigen.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – Ungültiger ARN -Servicefall

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid ARN service case: Update the service name ${service} in the resource ARN to use
 all lowercase letters.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Update the service name ${service} in the resource ARN to use all
 lowercase letters."
```

Beheben des Fehlers

Das Service im Ressourcen-ARN muss mit den Spezifikationen (einschließlich Groß-/Kleinschreibung) für Servicepräfixe übereinstimmen. Informationen zum Anzeigen des Präfixes für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service und suchen das Präfix im ersten Satz.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – Ungültiger Bedingungsdatentyp

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Beheben des Fehlers

Der Wert im Bedingungsschlüssel-Wert-Paar muss mit dem Datentyp des Bedingungsschlüssels und des Bedingungsoperators übereinstimmen. Informationen zum Datentyp des Bedingungsschlüssels für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service, um die Bedingungsschlüssel für diesen Service anzuzeigen.

Der globale Bedingungsschlüssel `CurrentTime` unterstützt zum Beispiel den `Date`-Bedingungsoperator. Wenn Sie eine Zeichenfolge oder eine Ganzzahl für den Wert im Bedingungsblock angeben, stimmt der Datentyp nicht überein.

Verwandte Begriffe

- Bedingungen
- Bedingungsblock
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- Globale Bedingungsschlüssel
- AWS Schlüssel für Servicebedingungen

Fehler – Ungültiges Bedingungsschlüsselformat

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Beheben des Fehlers

Der Schlüssel im Zustands-Schlüssel-Wert-Paar muss mit den Spezifikationen für das Service übereinstimmen. Informationen zum Anzeigen der Bedingungsschlüssel für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service, um die Bedingungsschlüssel für diesen Service anzuzeigen.

Verwandte Begriffe

- Bedingungen
- Globale Bedingungsschlüssel
- AWS Zustandsschlüssel für Dienste

Fehler – Ungültige Bedingung mehrere boolesche Werte

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Beheben des Fehlers

Der Schlüssel im Bedingungsschlüssel-Wert-Paar erwartet einen einzelnen booleschen Wert. Wenn Sie mehrere boolesche Werte angeben, gibt die Bedingungsübereinstimmung möglicherweise nicht die erwarteten Ergebnisse zurück.

Informationen zum Anzeigen der Bedingungsschlüssel für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service, um die Bedingungsschlüssel für diesen Service anzuzeigen.

- Bedingungen
- Globale Bedingungsschlüssel

- AWS Zustandsschlüssel für Dienste

Fehler – Ungültiger Bedingungsoperator

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Beheben des Fehlers

Aktualisieren Sie die Bedingung, um einen unterstützten Bedingungsoperator zu verwenden.

Verwandte Begriffe

- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- Condition-Element
- Übersicht über JSON-Richtlinien

Fehler – Ungültiger Effekt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Beheben des Fehlers

Aktualisieren des `Effect`-Element, um einen gültigen Effekt zu verwenden. Gültige Werte für `Effect` sind **`Allow`** und **`Deny`**.

Verwandte Begriffe

- Auswirkung
- Übersicht über JSON-Richtlinien

Fehler – Ungültiger -Bedingungsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Beheben des Fehlers

Aktualisieren Sie den Bedingungsschlüssel im Bedingungsschlüssel-Wert-Paar, um einen unterstützten globalen Bedingungsschlüssel zu verwenden.

Globale Bedingungsschlüssel sind Bedingungsschlüssel mit einem `aws:` Präfix. AWS Dienste können globale Bedingungsschlüssel unterstützen oder dienstspezifische Schlüssel bereitstellen, die ihr Dienstpräfix enthalten. Beispielsweise enthalten IAM-Bedingungsschlüssel das `iam:`-Präfix. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie dort den Dienst aus, dessen Schlüssel Sie anzeigen möchten.

Verwandte Begriffe

- Globale Bedingungsschlüssel

Fehler – Ungültige Partition

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Beheben des Fehlers

Aktualisieren Sie den Ressourcen-ARN, um eine unterstützte Partition einzuschließen. Wenn Sie eine unterstützte Partition enthalten, unterstützt das Service oder die Ressource die Partition, die Sie enthalten haben, möglicherweise nicht.

Eine Partition ist eine Gruppe von AWS Regionen. Jedes AWS Konto ist auf eine Partition beschränkt. Verwenden Sie in klassischen Regionen die `aws`-Partition. Verwenden Sie in China-Regionen `aws-cn`.

Verwandte Begriffe

- Amazon-Ressourcennamen (ARNs) - Partitionen

Fehler – Ungültiges Richtlinienelement

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid policy element: The policy element {{element}} is not valid.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The policy element {{element}} is not valid."
```

Beheben des Fehlers

Aktualisieren Sie die Richtlinie, um nur unterstützte JSON-Richtlinienelemente einzuschließen.

Verwandte Begriffe

- JSON-Richtlinienelemente

Fehler – Ungültiges Prinzipalformat

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

Beheben des Fehlers

Aktualisieren Sie den Prinzipal, um ein unterstütztes Schlüssel-Wert-Paarformat zu verwenden.

Sie können in einer ressourcenbasierten Richtlinie einen Prinzipal angeben, aber keine identitätsbasierte Richtlinie.

Um beispielsweise den Zugriff für alle Benutzer eines AWS Kontos zu definieren, verwenden Sie in Ihrer Richtlinie den folgenden Grundsatz:

```
"Principal": { "AWS": "123456789012" }
```

Verwandte Begriffe

- JSON-Richtlinienelemente: Prinzipal
- Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien

Fehler – Ungültiger Prinzipalschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Beheben des Fehlers

Aktualisieren Sie den Schlüssel im Hauptschlüssel-Wert-Paar, um einen unterstützten Prinzipalschlüssel zu verwenden. Die folgenden Hauptschlüssel werden unterstützt:

- AWS
- CanonicalUser
- Verbund
- Service

Verwandte Begriffe

- Prinzipal-Element

Fehler – Ungültiger Bereich

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 suported Region.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 suported Region."
```

Beheben des Fehlers

Aktualisieren Sie den Wert des Bedingungsschlüssel-Wert-Paars, um eine unterstützte Region einzuschließen. Eine Tabelle der in den einzelnen Regionen unterstützten AWS Dienste finden Sie in der Tabelle mit den Regionen.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- Regionsnamen und -codes

Fehler – Ungültiges Service

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Beheben des Fehlers

Das Servicepräfix im Aktions- oder Bedingungsschlüssel muss mit den Spezifikationen (einschließlich Groß-/Kleinschreibung) für Servicepräfixe übereinstimmen. Informationen zum Anzeigen des Präfixes für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service und suchen das Präfix im ersten Satz.

Verwandte Begriffe

- Bekannte Services und ihre Aktionen, Ressourcen und Bedingungsschlüssel

Fehler – Ungültiger Service-Bedingungsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Beheben des Fehlers

Aktualisieren Sie den Schlüssel im Bedingungsschlüssel-Wert-Paar, um einen bekannten Bedingungsschlüssel für das Service zu verwenden. Globale Bedingungsschlüssel sind Bedingungsschlüssel mit einem `aws`-Präfix. AWS -Services können servicespezifische Schlüssel bereitstellen, die das Service-Präfix enthalten. Informationen zum Anzeigen des Präfixes für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- Bekannte Services und ihre Aktionen, Ressourcen und Bedingungsschlüssel

Fehler – Ungültiges Service in Aktion

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Beheben des Fehlers

Das Servicepräfix in der Aktion muss mit den Spezifikationen (einschließlich Groß-/Kleinschreibung) für Servicepräfixe übereinstimmen. Informationen zum Anzeigen des Präfixes für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service und suchen das Präfix im ersten Satz.

Verwandte Begriffe

- Action-Element

- Bekannte Services und ihre Handlungen

Fehler – Ungültige Variable für den Operator

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Beheben des Fehlers

Sie können Richtlinienvariablen im `Resource`-Element und in Zeichenfolgenvergleichen im `Condition`-Element verwenden. Bedingungen unterstützen Variablen, wenn Sie String-Operatoren oder ARN Operatoren verwenden. String-Operatoren beinhalten `StringEquals`, `StringLike` und `StringNotLike`. ARN-Operatoren beinhalten `ArnEquals` und `ArnLike`. Sie können eine Richtlinienvariable nicht mit anderen Operatoren verwenden, z. B. mit numerischen, Datums-, booleschen, binären, IP-Adress- oder Null-Operatoren.

Verwandte Begriffe

- Verwenden von Richtlinienvariablen im Condition-Element
- Condition-Element

Ungültige Version

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid version: The version ${version} is not valid. Use one of the following
 versions: ${versions}
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The version ${version} is not valid. Use one of the following
 versions: ${versions}"
```

Beheben des Fehlers

Das `Version` Policy-Element gibt die Sprachsyntaxregeln an, die zur Verarbeitung einer Richtlinie AWS verwendet werden. Um alle verfügbaren Richtlinienfunktionen zu nutzen, fügen Sie das folgende Element `Version` vor dem Element `Statement` in alle Ihre Richtlinien ein.

```
"Version": "2012-10-17"
```

Verwandte Begriffe

- Element der Version

Fehler – Json-Syntaxfehler

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Beheben des Fehlers

Ihre Richtlinie enthält einen Syntaxfehler. Überprüfen Sie Ihre JSON-Syntax.

Verwandte Begriffe

- JSON-Validator
- IAM-JSON-Richtlinienelementreferenz
- Übersicht über JSON-Richtlinien

Fehler – Json-Syntaxfehler

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Json syntax error: Fix the JSON syntax error.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Fix the JSON syntax error."
```

Beheben des Fehlers

Ihre Richtlinie enthält einen Syntaxfehler. Überprüfen Sie Ihre JSON-Syntax.

Verwandte Begriffe

- JSON-Validator
- IAM-JSON-Richtlinienelementreferenz
- Übersicht über JSON-Richtlinien

Fehler – Fehlende Aktion

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing action: Add an Action or NotAction element to the policy statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Beheben des Fehlers

AWS JSON-Richtlinien müssen ein `Action` `NotAction` OR-Element enthalten.

Verwandte Begriffe

- Action-Element

- NotAction Element
- Übersicht über JSON-Richtlinien

Fehler – Fehlendes ARN Feld

In dem AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Beheben des Fehlers

Alle Felder im Ressourcen-ARN müssen mit den Spezifikationen für einen bekannten Ressourcentyp übereinstimmen. Informationen zum erwarteten ARN-Format für einen Dienst finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie den Namen des Service aus, um seine Ressourcentypen und ARN Formate anzuzeigen.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- AWS Serviceressourcen mit ARN-Formaten

Fehler – Fehlende ARN Region

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Beheben des Fehlers

Die Ressourcen-ARNs für die meisten AWS Dienste erfordern, dass Sie eine Region angeben. Eine Tabelle der in den einzelnen Regionen unterstützten AWS Dienste finden Sie in der Regionstabelle.

Verwandte Begriffe

- Richtlinienressourcen
- Ressourcen-ARNs
- Regionsnamen und -codes

Fehler – Fehlender Effekt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Beheben des Fehlers

AWS JSON-Richtlinien müssen ein `Effect` Element mit dem Wert **`Allow`** und **`Deny`** enthalten.

Verwandte Begriffe

- Auswirkung
- Übersicht über JSON-Richtlinien

Fehler – Fehlender Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing principal: Add a Principal element to the policy statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a Principal element to the policy statement."
```

Beheben des Fehlers

Ressourcenbasierte Richtlinien müssen einen `Principal`-Element.

Um beispielsweise den Zugriff für alle Benutzer eines AWS Kontos zu definieren, verwenden Sie in Ihrer Richtlinie den folgenden Grundsatz:

```
"Principal": { "AWS": "123456789012" }
```

Verwandte Begriffe

- Prinzipal-Element
- Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien

Fehler – Fehlender Qualifizierer

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing qualifier: The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Beheben des Fehlers

Im `Condition`-Element formulieren Sie Ausdrücke, in denen Sie Bedingungsoperatoren verwenden (gleich, kleiner als usw.), um die Bedingungsschlüssel und -werte der Richtlinie mit den Schlüsseln

und Werten in der Anforderungen abzugleichen. Bei Anforderungen, die mehrere Werte für einen einzelnen Schlüssel enthalten, müssen Sie die Bedingungen in Klammern, wie in einem Array ("Key2": ["Value2A", "Value2B"]), einschließen. Sie müssen zudem die Satzoperatoren `ForAllValues` oder `ForAnyValue` mit dem `StringLike` Bedingungsoperator verwenden. Diese Qualifizierer fügen die Set-Operationen zum Bedingungsoperator hinzu, sodass Sie mehrere Anforderungswerte mit mehreren Bedingungswerten testen können.

Verwandte Begriffe

- Mehrwertige Kontextschlüssel
- Condition-Element

AWS verwaltete Richtlinien mit diesem Fehler

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Bei den folgenden AWS verwalteten Richtlinien fehlt in ihren Richtlinienerklärungen ein Kennzeichner für Bedingungsschlüssel. Wenn Sie die AWS verwaltete Richtlinie als Referenz für die Erstellung Ihrer vom Kunden verwalteten Richtlinie verwenden, AWS empfiehlt Ihnen, Ihrem `Condition` Element die Schlüsselqualifikatoren `ForAllValues` oder die `ForAnyValue` Bedingung hinzuzufügen.

- AWSGlueConsoleSageMakerNotebookFullAccess

Fehler – Fehlende Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Beheben des Fehlers

Alle Richtlinien mit Ausnahme der Richtlinien zur Rollenvertrauensstellung müssen ein `Resource` `NotResource` Oder-Element enthalten.

Verwandte Begriffe

- Ressourcen-Element
- NotResource Element
- Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien
- Übersicht über JSON-Richtlinien

Fehler – Fehlende Anweisung

In dem AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing statement: Add a statement to the policy
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a statement to the policy"
```

Beheben des Fehlers

Eine JSON-Richtlinie muss eine Anweisung enthalten.

Verwandte Begriffe

- JSON-Richtlinienelemente

Fehler – Null mit, wenn vorhanden

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Beheben des Fehlers

Sie können `IfExists` an das Ende des Namens eines beliebigen Bedingungsoperators mit Ausnahme des Bedingungsoperators `Null` anfügen. Verwenden Sie den Bedingungsoperator `Null`, um zu prüfen, ob ein Bedingungsschlüssel zum Zeitpunkt der Autorisierung vorhanden ist. Verwenden Sie `...ifExists`, um zu sagen: "Wenn der Richtlinienschlüssel im Kontext der Anforderung vorhanden ist, verarbeiten Sie den Schlüssel wie in der Richtlinie angegeben. Wenn der Schlüssel nicht vorhanden ist, wird das Bedingungselement als "true" ausgewertet.

Verwandte Begriffe

- ... IfExists Bedingungsoperatoren
- Null-Bedingungs-Operator
- Condition-Element

Fehler – SCP-Syntaxfehleraktion Platzhalter

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen die Angabe von Werten in den `Action` `NotAction` OR-Elementen. Diese Werte können jedoch Platzhalter (*) nur am Ende der Zeichenfolge enthalten. Dies bedeutet, dass Sie `iam:Get*` aber nicht `iam:*role` auswählen können.

Um mehrere Aktionen anzugeben, AWS empfiehlt es sich, diese einzeln aufzulisten.

Verwandte Begriffe

- SCP-Aktion und Elemente NotAction
- SCP-Bewertung
- AWS Organizations Richtlinien zur Servicesteuerung
- IAM-JSON-Richtlinienelemente: Aktion

Fehler – SCP-Syntaxfehler erlaubt Bedingung

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
SCP syntax error allow condition: SCPs do not support the Condition element with effect
 Allow. Update the element Condition or the effect.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "SCPs do not support the Condition element with effect Allow. Update
 the element Condition or the effect."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen die Angabe von Werten im `Condition` Element nur, wenn Sie Folgendes verwenden. `"Effect": "Deny"`

Um nur eine einzige Aktion zuzulassen, können Sie den Zugriff auf alles außer der Bedingung verweigern, die Sie mit der `...NotEquals`-Version eines Bedingungsoperators angeben. Dies negiert den Vergleich des Operators.

Verwandte Begriffe

- SCP Konditionselement
- SCP-Bewertung
- AWS Organizations Richtlinien zur Servicesteuerung
- Beispielrichtlinie: Verweigert den Zugriff auf AWS basierend auf der angeforderten Region
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

- IAM-JSON-Richtlinienelemente: Condition

Fehler — SCP-Syntaxfehler zulassen NotAction

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
SCP syntax error allow NotAction: SCPs do not support NotAction with effect Allow.
 Update the element NotAction or the effect.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "SCPs do not support NotAction with effect Allow. Update the element
 NotAction or the effect."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen die Verwendung des `NotAction` Elements with nicht. `"Effect": "Allow"`

Sie müssen die Logik neu schreiben, um eine Liste von Aktionen zuzulassen oder jede Aktion abzulehnen, die nicht aufgeführt ist.

Verwandte Begriffe

- SCP-Aktion und Elemente NotAction
- SCP-Bewertung
- AWS Organizations Richtlinien zur Servicesteuerung
- IAM-JSON-Richtlinienelemente: Aktion

Fehler – SCP-Syntaxfehler erlaubt Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
SCP syntax error allow resource: SCPs do not support Resource with effect Allow. Update
 the element Resource or the effect.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "SCPs do not support Resource with effect Allow. Update the element
 Resource or the effect."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen die Angabe von Werten im `Resource` Element nur, wenn Sie Folgendes verwenden. `"Effect": "Deny"`

Sie müssen die Logik neu schreiben, um alle Ressourcen zuzulassen oder jede aufgelistete Ressource zu verweigern.

Verwandte Begriffe

- SCP Ressourcenelement
- SCP-Bewertung
- AWS Organizations Richtlinien zur Servicesteuerung
- IAM-JSON-Richtlinienelemente: Resource

Fehler — SCP-Syntaxfehler NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
SCP syntax error NotResource: SCPs do not support the NotResource element. Update the
 policy to use Resource instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "SCPs do not support the NotResource element. Update the policy to
 use Resource instead."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen das `NotResource` Element nicht.

Sie müssen die Logik neu schreiben, um alle Ressourcen zuzulassen oder jede aufgelistete Ressource zu verweigern.

Verwandte Begriffe

- SCP Ressourcenelement
- SCP-Bewertung
- AWS Organizations Richtlinien zur Dienstkontrolle
- IAM-JSON-Richtlinienelemente: Ressource

Fehler – SCP-Syntaxfehler-Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen die Elemente `Principal` oder `NotPrincipal` nicht.

Sie können den Amazon Resource Name (ARN) über den `aws:PrincipalArn`-globalen Bedingungsschlüssel im `Condition`-Element angeben.

Verwandte Begriffe

- SCP-Syntax
- Allgemeine Bedingungsschlüssel für Prinzipal

Fehler – Eindeutige Sids erforderlich

Das AWS Management Console Ergebnis dieser Prüfung beinhaltet die folgende Meldung:

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Beheben des Fehlers

Bei einigen Richtlinientypen müssen Anweisungs-IDs eindeutig sein. Die `Sid` (Statement-ID) ist eine optionale ID, die Sie in die Richtlinie aufnehmen können. Sie können jeder Anweisung in einem Anweisungsarray einen Anweisungs-ID-Wert zuweisen, indem Sie das `SID`-Element verwenden. Bei Services, die die Angabe eines ID-Elements zulassen, wie z. B. SQS und SNS, ist der Wert `Sid` lediglich eine Sub-ID der ID des Richtliniendokuments. In IAM muss der `Sid`-Wert innerhalb einer JSON-Richtlinie eindeutig sein.

Verwandte Begriffe

- IAM-JSON-Richtlinienelemente: Sid

Fehler - Nicht unterstützte Aktion in der Richtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Beheben des Fehlers

Einige Aktionen werden im `Action`-Element in der ressourcenbasierten Richtlinie nicht unterstützt, die an einen anderen Ressourcentyp gebunden sind. Beispielsweise werden AWS Key Management Service Aktionen in Amazon S3 S3-Bucket-Richtlinien nicht unterstützt. Geben Sie eine Aktion an, die vom Ressourcentyp unterstützt wird, der an Ihre ressourcenbasierte Richtlinie angehängt ist.

Verwandte Begriffe

- JSON-Richtlinienelemente: Aktion

Fehler – Nicht unterstützte Elementkombination

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported element combination: The policy elements ${element1} and ${element2} can
 not be used in the same statement. Remove one of these elements.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The policy elements ${element1} and ${element2} can not be used in
 the same statement. Remove one of these elements."
```

Beheben des Fehlers

Einige Kombinationen von JSON-Richtlinienelementen können nicht zusammen verwendet werden. Sie können beispielsweise `Action` und `NotAction` nicht in der gleichen Richtlinienanweisung verwenden. Weitere Paare, die sich gegenseitig ausschließen, sind `Principal/NotPrincipal` und `Resource/NotResource`.

Verwandte Begriffe

- IAM-JSON-Richtlinienelementreferenz
- Übersicht über JSON-Richtlinien

Fehler – Nicht unterstützter globaler -Bedingungsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Beheben des Fehlers

AWS unterstützt die Verwendung des angegebenen globalen Bedingungsschlüssels nicht. Je nach Anwendungsfall können Sie den `aws:PrincipalArn` oder `aws:SourceArn` globale Bedingungsschlüssel. Verwenden Sie z. B. `aws:ARN` anstelle von `aws:PrincipalArn`, um den Amazon Resource Name (ARN) des Principals, der die Anforderung gestellt hat, mit dem ARN zu vergleichen, den Sie in der Richtlinie angeben. Verwenden Sie alternativ den `aws:SourceArn` globalen Bedingungsschlüssel, um den Amazon-Ressourcennamen (ARN) der Ressource, die eine service-to-service Anfrage stellt, mit dem ARN zu vergleichen, den Sie in der Richtlinie angeben.

Verwandte Begriffe

- AWS Kontextschlüssel für globale Bedingungen

Fehler – Nicht unterstützter Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported principal: The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element."
```

Beheben des Fehlers

Verwenden Sie das `Principal`-Element in einer Richtlinie, um den Prinzipal anzugeben, dem der Zugriff auf eine Ressource erlaubt oder verweigert wird. Sie können das Element `Principal` nicht in einer identitätsbasierten IAM-Richtlinie verwenden. Sie können es in den Vertrauensrichtlinien für IAM-Rollen und in ressourcenbasierten Richtlinien verwenden. Ressourcenbasierte Richtlinien sind Richtlinien, die Sie direkt in eine Ressource einbinden. Sie können beispielsweise Richtlinien in einen Amazon S3 S3-Bucket oder einen AWS KMS-Schlüssel einbetten.

Verwandte Begriffe

- AWS JSON-Richtlinienelemente: Prinzipal
- Kontoübergreifender Zugriff auf Ressourcen in IAM

Fehler - Nicht unterstützter Ressourcen-ARN in der Richtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Beheben des Fehlers

Einige Ressourcen-ARNs werden im `Resource`-Element der ressourcenbasierten Richtlinie nicht unterstützt, wenn die Richtlinie an einen anderen Ressourcentyp angehängt wird. Beispielsweise werden AWS KMS ARNs im `Resource` Element für Amazon S3 S3-Bucket-Richtlinien nicht unterstützt. Geben Sie einen Ressourcen-ARN an, der von einem Ressourcentyp unterstützt wird, der an Ihre ressourcenbasierte Richtlinie angehängt ist.

Verwandte Begriffe

- JSON-Richtlinienelemente: Aktion

Fehler – Nicht unterstützte Sid

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported Sid: Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Beheben des Fehlers

Das `Sid`-Element unterstützt Großbuchstaben, Kleinbuchstaben und Zahlen.

Verwandte Begriffe

- IAM-JSON-Richtlinienelemente: Sid

Fehler – Nicht unterstützter Platzhalter im Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Beheben des Fehlers

Die`Principal`-Elementstruktur unterstützt die Verwendung eines Schlüssel-Wert-Paares. Der in der Richtlinie angegebene Hauptwert enthält einen Platzhalter (*). Sie können keinen Platzhalter mit dem angegebenen Hauptschlüssel einschließen. Wenn Sie z. B. Benutzer in einem `Principal`-Element angeben, können Sie nicht mit einem Platzhalter "alle Benutzer" meinen. Benennen Sie stattdessen einen oder mehrere bestimmte Benutzer. Wenn Sie eine Sitzung mit übernommener Rolle in einem -Element angeben, können Sie keinen Platzhalter (*) verwenden, der "alle Sitzungen" bedeutet. Sie müssen eine bestimmte Sitzung benennen. Sie können keinen Platzhalter verwenden, um einen Teil eines Namens oder einen ARN zu ersetzen.

Um dieses Ergebnis zu beheben, entfernen Sie den Platzhalter und geben Sie einen spezifischeren Prinzipal an.

Verwandte Begriffe

- AWS JSON-Richtlinienelemente: Prinzipal

Fehler – Fehlende Klammer in Variable

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Beheben des Fehlers

Die Struktur von Richtlinienvariablen unterstützt die Verwendung eines $-Präfixes, gefolgt von einem Paar geschweifter Klammern ({ }). Innerhalb der ${ }-Zeichen können Sie den Namen des Wertes aus der Anforderung einfügen, den Sie in der Richtlinie verwenden möchten.

Um dieses Ergebnis zu beheben, fügen Sie die fehlende Klammer hinzu, um sicherzustellen, dass der vollständige Öffnungs- und Schließungssatz von geschweiften Klammern vorhanden ist.

Verwandte Begriffe

- IAM-Richtlinienelemente: Variablen

Fehler – Fehlendes Zitat in Variable

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Beheben des Fehlers

Wenn Sie Ihrer Richtlinie eine Variable hinzufügen, können Sie einen Standardwert für die Variable angeben. Wenn eine Variable nicht vorhanden ist, wird der von Ihnen angegebene Standardtext AWS verwendet.

Um einer Variablen einen Standardwert hinzuzufügen, umgeben Sie den Standardwert mit einfachen Anführungszeichen (`' '`), und trennen Sie den Variablentext und den Standardwert durch Komma und Leerzeichen (`, `) enthalten.

Zum Beispiel, wenn ein Prinzipal mit `team=yellow` gekennzeichnet ist, können sie auf `DOC-EXAMPLE-BUCKET` der Amazon-S3-Bucket mit dem Namen `DOC-EXAMPLE-BUCKET-yellow` zugreifen. Eine Richtlinie mit dieser Ressource ermöglicht es Teammitgliedern möglicherweise auf ihre eigenen Ressourcen zuzugreifen, aber nicht auf die anderer Teams. Für Benutzer ohne Team-Tags können Sie den Standardwert von `company-wide` einstellen. Diese Benutzer können nur auf `DOC-EXAMPLE-BUCKET-company-wide`-Bucket zugreifen, indem sie umfassende Informationen anzeigen können, z. B. Anweisungen für den Beitritt zu einem Team.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Verwandte Begriffe

- IAM-Richtlinienelemente: Variablen

Fehler – Nicht unterstützter Speicherplatz in der Variablen

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Beheben des Fehlers

Die Struktur von Richtlinienvariablen unterstützt die Verwendung eines $-Präfixes, gefolgt von einem Paar geschweifter Klammern ({ }). Innerhalb der ${ }-Zeichen können Sie den Namen des Wertes aus der Anforderung einfügen, den Sie in der Richtlinie verwenden möchten. Obwohl Sie ein Leerzeichen hinzufügen können, wenn Sie eine Standardvariable angeben, können Sie kein Leerzeichen in den Variablennamen einschließen.

Verwandte Begriffe

- IAM-Richtlinienelemente: Variablen

Fehler – Leere Variable

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Beheben des Fehlers

Die Struktur von Richtlinienvariablen unterstützt die Verwendung eines $-Präfixes, gefolgt von einem Paar geschweifter Klammern ({ }). Innerhalb der ${ }-Zeichen können Sie den Namen des Wertes aus der Anforderung einfügen, den Sie in der Richtlinie verwenden möchten.

Verwandte Begriffe

- IAM-Richtlinienelemente: Variablen

Fehler – Variable im Element nicht unterstützt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Beheben des Fehlers

Sie können Richtlinienvariablen im `Resource`-Element und in Zeichenfolgenvergleichen im `Condition`-Element verwenden.

Verwandte Begriffe

- IAM-Richtlinienelemente: Variablen

Fehler – Variable wird in Version nicht unterstützt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Beheben des Fehlers

Zur Verwendung der Richtlinienvariablen müssen Sie das `Version`-Element in eine Anweisung einfügen und die Version muss auf eine Version gesetzt werden, die Richtlinienvariablen unterstützt. Variablen wurden in Version `2012-10-17` eingeführt. Frühere Versionen der Richtliniensprache unterstützen Richtlinienvariablen nicht. Wenn Sie `Version` nicht auf `2012-10-17` oder später setzen, werden Variablen wie `${aws:username}` in der Richtlinie als literale Zeichenketten behandelt.

Das Richtlinienelement `Version` unterscheidet sich von einer Richtlinienversion. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Eine Richtlinienversion hingegen wird erstellt, wenn Sie in IAM eine

kundenverwaltete Richtlinie bearbeiten. Die vorhandene Richtlinie wird von der geänderten Richtlinie nicht überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie.

Verwandte Begriffe

- IAM-Richtlinienelemente: Variablen
- IAM JSON-Richtlinienelemente: Version

Private IP-Adresse

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Beheben des Fehlers

Der globale Bedingungsschlüssel `aws:SourceIp` funktioniert nur für öffentliche IP-Adressbereiche. Sie erhalten diesen Fehler, wenn Ihre Richtlinie nur private IP-Adressen zulässt. In diesem Fall würde die Bedingung niemals übereinstimmen.

- aws: SourceIp globaler Bedingungsschlüssel
- IAM-JSON-Richtlinienelemente: Condition

Fehler — Privat NotIpAddress

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Beheben des Fehlers

Der globale Bedingungsschlüssel `aws:SourceIp` funktioniert nur für öffentliche IP-Adressbereiche. Sie erhalten diesen Fehler, wenn Sie die `NotIpAddress`-Bedingungsoperator und listet nur private IP-Adressen auf. In diesem Fall würde die Bedingung immer übereinstimmen und wäre unwirksam.

- aws: SourceIp globaler Bedingungsschlüssel
- IAM-JSON-Richtlinienelemente: Condition

Fehler – Richtliniengröße überschreitet das SCP-Kontingent

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Beheben des Fehlers

AWS Organizations Service Control Policies (SCPs) unterstützen die Angabe von Werten in den `Action` `NotAction` OR-Elementen. Diese Werte können jedoch Platzhalter (*) nur am Ende der Zeichenfolge enthalten. Dies bedeutet, dass Sie `iam:Get*` aber nicht `iam:*role` auswählen können.

Um mehrere Aktionen anzugeben, AWS empfiehlt es sich, diese einzeln aufzulisten.

Verwandte Begriffe

- Kontingente für AWS Organizations
- AWS Organizations Richtlinien zur Servicesteuerung

Fehler – Ungültiges Service-Prinzipalformat

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Beheben des Fehlers

Der Wert im Bedingungsschlüssel-Wert-Paar muss mit einem definierten Service-Prinzipalformat übereinstimmen.

Ein Service-Prinzipal ist eine Kennung, die verwendet wird, um einem Service Berechtigungen zu erteilen. Sie können einen Service-Prinzipal im `Principal`-Element oder einen Wert für einige globale Bedingungsschlüssel und servicespezifische Schlüssel angeben. Der Service-Prinzipal wird durch den Service definiert.

Der Bezeichner für einen Service-Prinzipal enthält den Servicenamen und hat normalerweise das folgende Format in Kleinbuchstaben:

`service-name``.amazonaws.com`

Einige servicespezifische Schlüssel verwenden möglicherweise ein anderes Format für Service-Prinzipal. Zum Beispiel, der `kms:ViaService`-Bedingungsschlüssel erfordert das folgende Format für Service-Prinzipal in Kleinbuchstaben:

`service-name.AWS_region``.amazonaws.com`

Verwandte Begriffe

- Service-Prinzipal
- AWS globale Bedingungsschlüssel
- `kms:ViaService` Bedingungsschlüssel

Fehler – Fehlender Tag-Schlüssel in Bedingung

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Beheben des Fehlers

Um den Zugriff auf Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an.

Um beispielsweise den Zugriff auf AWS Ressourcen zu steuern, fügen Sie den `aws:ResourceTag` Bedingungsschlüssel hinzu. Dieser Schlüssel benötigt das Format `aws:ResourceTag/`*`tag-key`*. Um den Tag-Schlüssel `owner` und den Tag-Wert `JaneDoe` in einer Bedingung anzugeben, verwenden Sie das folgende Format.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

Verwandte Begriffe

- Zugriffssteuerung mit Tags
- Bedingungen
- Globale Bedingungsschlüssel

- AWS Schlüssel für Servicebedingungen

Fehler - Ungültiges vpc-Format

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Beheben des Fehlers

Der `aws:SourceVpc`-Bedingungsschlüssel muss das Präfix `vpc-` verwenden, gefolgt von entweder 8 oder 17 alphanumerischen Zeichen, z. B. `vpc-11223344556677889` oder `vpc-12345678`.

Verwandte Begriffe

- AWS globale Bedingungsschlüssel: aws: SourceVpc

Fehler - Ungültiges vpce-Format

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Beheben des Fehlers

Der `aws:SourceVpce`-Bedingungsschlüssel muss das Präfix `vpce-` verwenden, gefolgt von entweder 8 oder 17 alphanumerischen Zeichen, z. B. `vpce-11223344556677889` oder `vpce-12345678`.

Verwandte Begriffe

- AWS globale Bedingungsschlüssel: aws: SourceVpce

Fehler - Verbundprinzipal wird nicht unterstützt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Beheben des Fehlers

Das `Principal`-Element verwendet Verbundprinzipale für Vertrauensrichtlinien, die an IAM-Rollen angehängt sind, um Zugriff über den Identitätsverbund zu ermöglichen. Identitätsrichtlinien und andere ressourcenbasierte Richtlinien unterstützen keinen Verbundidentitätsanbieter im `Principal`-Element. Sie können beispielsweise keinen SAML-Prinzipal in einer Amazon-S3-Bucket-Richtlinie verwenden. Ändern Sie das `Principal`-Element zu einem unterstützten Prinzipaltyp.

Verwandte Begriffe

- Erstellen einer Rolle für den Identitätsverbund
- JSON-Richtlinienelemente: Prinzipal

Fehler - Nicht unterstützte Aktion für Bedingungsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
```

```
actions. We recommend that you move these actions to a different statement without
this condition key.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
 recommend that you move these actions to a different statement without this condition
 key."
```

Beheben des Fehlers

Stellen Sie sicher, dass der Bedingungsschlüssel im `Condition`-Element der Richtlinienanweisung für jede Aktion im `Action`-Element gilt. Um sicherzustellen, dass die von Ihnen angegebenen Aktionen von Ihrer Richtlinie effektiv zulässig oder verweigert werden, sollten Sie die nicht unterstützten Aktionen ohne den Bedingungsschlüssel in eine andere Anweisung verschieben.

> **Note**
>
> Wenn das `Action`-Element Aktionen mit Platzhaltern hat, wertet IAM Access Analyzer diese Aktionen nicht auf diesen Fehler aus.

Verwandte Begriffe

- JSON-Richtlinienelemente: Aktion

Fehler - Nicht unterstützte Aktion in der Richtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Beheben des Fehlers

Einige Aktionen werden im `Action`-Element in der ressourcenbasierten Richtlinie nicht unterstützt, die an einen anderen Ressourcentyp gebunden sind. Beispielsweise werden AWS Key Management Service Aktionen in Amazon S3 S3-Bucket-Richtlinien nicht unterstützt. Geben Sie eine Aktion an, die vom Ressourcentyp unterstützt wird, der an Ihre ressourcenbasierte Richtlinie angehängt ist.

Verwandte Begriffe

- JSON-Richtlinienelemente: Aktion

Fehler - Nicht unterstützter Ressourcen-ARN in der Richtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Beheben des Fehlers

Einige Ressourcen-ARNs werden im `Resource`-Element der ressourcenbasierten Richtlinie nicht unterstützt, wenn die Richtlinie an einen anderen Ressourcentyp angehängt wird. Beispielsweise werden AWS KMS ARNs im `Resource` Element für Amazon S3 S3-Bucket-Richtlinien nicht unterstützt. Geben Sie einen Ressourcen-ARN an, der von einem Ressourcentyp unterstützt wird, der an Ihre ressourcenbasierte Richtlinie angehängt ist.

Verwandte Begriffe

- JSON-Richtlinienelemente: Aktion

Fehler - Nicht unterstützter Bedingungsschlüssel für Service-Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Beheben des Fehlers

Sie können AWS-Services im `Principal` Element einer ressourcenbasierten Richtlinie einen Dienstprinzipal angeben, bei dem es sich um einen Bezeichner für den Dienst handelt. Sie können einige Bedingungsschlüssel mit bestimmten Service-Prinzipalen nicht verwenden. Sie können beispielsweise den `aws:PrincipalOrgID`-Bedingungsschlüssel mit dem Service-Prinzipal `cloudfront.amazonaws.com` nicht verwenden. Sie sollten Bedingungsschlüssel entfernen, die nicht für den Service-Prinzipal im `Principal`-Element zutreffen.

Verwandte Begriffe

- Service-Prinzipal
- JSON-Richtlinienelemente: Prinzipal

Fehler – Syntaxfehler der Rollenvertrauensrichtlinie: notprincipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Beheben des Fehlers

Eine Rollenvertrauensrichtlinie ist eine ressourcenbasierte Richtlinie die einer IAM-Rolle angefügt ist. Vertrauensrichtlinien definieren, welche Prinzipal-Entitäten (Konten, Benutzer, Rollen und Verbundbenutzer) die Rolle übernehmen können. Rollenvertrauensrichtlinien unterstützen `NotPrincipal` nicht. Aktualisieren Sie die Richtlinie, um stattdessen ein`Principal`-Element zu verwenden.

Verwandte Begriffe

- JSON-Richtlinienelemente: Prinzipal
- JSON-Richtlinienelemente: NotPrincipal

Fehler – Rollenvertrauensrichtlinie – nicht unterstützter Platzhalter in Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Beheben des Fehlers

Eine Rollenvertrauensrichtlinie ist eine ressourcenbasierte Richtlinie die einer IAM-Rolle angefügt ist. Vertrauensrichtlinien definieren, welche Prinzipal-Entitäten (Konten, Benutzer, Rollen und Verbundbenutzer) die Rolle übernehmen können. `"Principal:" "*"` wird nicht im `Principal`-Element einer Rollenvertrauensrichtlinie unterstützt. Ersetzen Sie den Platzhalter durch einen gültigen Prinzipalwert.

Verwandte Begriffe

- JSON-Richtlinienelemente: Prinzipal

Fehler – Syntaxfehler der Rollenvertrauensrichtlinie: error resource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Beheben des Fehlers

Eine Rollenvertrauensrichtlinie ist eine ressourcenbasierte Richtlinie die einer IAM-Rolle angefügt ist. Vertrauensrichtlinien definieren, welche Prinzipal-Entitäten (Konten, Benutzer, Rollen und Verbundbenutzer) die Rolle übernehmen können. Rollenvertrauensrichtlinien gelten für die Rolle, mit der sie verknüpft sind. Sie können kein `Resource`- oder `NotResource`-Element in einer Rollenvertrauensrichtlinie angeben. Entfernen Sie das `Resource`- oder `NotResource`-Element.

- JSON-Richtlinienelemente: Resource
- JSON-Richtlinienelemente: NotResource

Fehler – Typ stimmt nicht mit IP-Bereich überein

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Beheben des Fehlers

Aktualisieren Sie den Text, um den Datentyp für den IP-Adressenbedingung im CIDR-Format zu verwenden.

Verwandte Begriffe

- Bedingungsoperatoren für IP-Adressen
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Fehler – Fehlende Aktion für Bedingungsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Beheben des Fehlers

Der Bedingungsschlüssel im `Condition`-Element der Richtlinienanweisung wird nicht ausgewertet, es sei denn, die angegebene Aktion befindet sich im `Action`-Element. Um sicherzustellen, dass die von Ihnen angegebenen Bedingungsschlüssel von Ihrer Richtlinie effektiv zugelassen oder verweigert werden, fügen Sie die Aktion zum `Action`-Element hinzu.

Verwandte Begriffe

- JSON-Richtlinienelemente: Aktion

Fehler – Ungültige Verbundprinzipal-Syntax in Rollenvertrauensrichtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Beheben des Fehlers

Der Prinzipalwert gibt einen Verbundprinzipal an, der nicht dem erwarteten Format entspricht. Aktualisieren Sie das Format des Verbundprinzipals in einen gültigen Domainnamen oder einen SAML-Metadaten-ARN.

Verwandte Begriffe

- Verbundbenutzer und -rollen

Fehler – Nicht übereinstimmende Aktion für Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Beheben des Fehlers

Die im `Action`-Element der Richtlinienanweisung angegebene Aktion ist ungültig, wobei der Prinzipal im `Principal`-Element angegeben ist. Sie können beispielsweise keinen SAML-

Anbieterprinzipal mit der `sts:AssumeRoleWithWebIdentity`-Aktion verwenden. Sie sollten einen SAML-Anbieterprinzipal mit der `sts:AssumeRoleWithSAML`-Aktion oder einen OIDC-Anbieterprinzipal mit der`sts:AssumeRoleWithWebIdentity`-Aktion verwenden.

Verwandte Begriffe

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

Fehler – Fehlende Aktion für die Vertrauensrichtlinie von Roles Anywhere

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Beheben des Fehlers

Damit IAM Roles Anywhere eine Rolle übernehmen und temporär AWS -Anmeldeinformationen bereitstellen kann, muss die Rolle dem Serviceprinzipal von IAM Roles Anywhere vertrauen. Der IAM Roles Anywhere-Serviceprinzipal benötigt `sts:AssumeRole`-,`sts:SetSourceIdentity`- und `sts:TagSession`-Berechtigungen zur Übernahme einer Rolle. Wenn eine der Berechtigungen fehlt, müssen Sie sie Ihrer Richtlinie hinzufügen.

Verwandte Begriffe

- Vertrauensmodell in AWS Identity and Access Management Roles Anywhere

Allgemeine Warnung — Erstellen Sie eine Spiegelreflexkamera mit NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Behebung der allgemeinen Warnung

Die Aktion `iam:CreateServiceLinkedRole` erteilt die Erlaubnis, eine IAM-Rolle zu erstellen, die es einem AWS Dienst ermöglicht, Aktionen in Ihrem Namen auszuführen. Die Verwendung von `iam:CreateServiceLinkedRole` in einer Richtlinie mit dem `NotResource`-Element kann dazu führen, dass unbeabsichtigte serviceverknüpfte Rollen für mehrere Ressourcen erstellt werden. AWS empfiehlt, stattdessen zulässige ARNs im `Resource`-Element anzugeben.

- CreateServiceLinkedRole Betrieb
- Elemente der IAM-JSON-Richtlinie: NotResource
- IAM-JSON-Richtlinienelemente: Resource

Allgemeine Warnung — Erstellen Sie eine Spiegelreflexkamera mit Star in Aktion und NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Behebung der allgemeinen Warnung

Die Aktion `iam:CreateServiceLinkedRole` erteilt die Erlaubnis, eine IAM-Rolle zu erstellen, die es einem AWS Dienst ermöglicht, Aktionen in Ihrem Namen auszuführen. Richtlinien mit einem Platzhalter (*) im Feld `Action` und dazu gehören die `NotResource`-Element kann das Erstellen von unbeabsichtigten serviceverknüpften Rollen für mehrere Ressourcen ermöglichen. AWS empfiehlt, dass Sie zulässige ARNs im `Resource`-Element.

- CreateServiceLinkedRole Betrieb
- Elemente der IAM-JSON-Richtlinie: NotResource
- IAM-JSON-Richtlinienelemente: Resource

Allgemeine Warnung — Erstellen Sie eine Spiegelreflexkamera mit und NotAction NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Behebung der allgemeinen Warnung

Die Aktion `iam:CreateServiceLinkedRole` erteilt die Erlaubnis, eine IAM-Rolle zu erstellen, die es einem AWS Dienst ermöglicht, Aktionen in Ihrem Namen auszuführen. Die Verwendung des `NotAction`-Elements mit dem `NotResource`-Element kann dazu führen, dass unbeabsichtigte serviceverknüpfte Rollen für mehrere Ressourcen erstellt werden. AWS empfiehlt, dass Sie die Richtlinie umschreiben, um stattdessen `iam:CreateServiceLinkedRole` auf einer begrenzten Liste von ARNs im `Resource`-Element zu erlauben. Sie können auch `iam:CreateServiceLinkedRole` zu dem Element `NotAction` hinzufügen.

- CreateServiceLinkedRole Betrieb

- Elemente der IAM-JSON-Richtlinie: NotAction
- IAM-JSON-Richtlinienelemente: Aktion
- IAM-JSON-Richtlinienelemente: NotResource
- IAM-JSON-Richtlinienelemente: Resource

Allgemeine Warnung – Erstellen Sie SLR mit Stern in der Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Behebung der allgemeinen Warnung

Die Aktion `iam:CreateServiceLinkedRole` erteilt die Erlaubnis, eine IAM-Rolle zu erstellen, die es einem AWS Dienst ermöglicht, Aktionen in Ihrem Namen auszuführen. Die Verwendung von `iam:CreateServiceLinkedRole` in einer Richtlinie mit einem Platzhalter (*) im `Resource`-Element kann dazu führen, dass unbeabsichtigte serviceverknüpfte Rollen für mehrere Ressourcen erstellt werden. AWS empfiehlt, dass Sie stattdessen zulässige ARNs im `Resource`-Element angeben.

- CreateServiceLinkedRole Betrieb
- IAM-JSON-Richtlinienelemente: Resource

AWS verwaltete Richtlinien mit dieser allgemeinen Warnung

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Einige dieser Anwendungsfälle sind für Hauptbenutzer in Ihrem Konto bestimmt. Die folgenden AWS verwalteten Richtlinien bieten Hauptbenutzerzugriff und gewähren Berechtigungen zum Erstellen von dienstbezogenen Rollen für jeden AWS Dienst. AWS empfiehlt, dass Sie die folgenden AWS verwalteten Richtlinien nur IAM-Identitäten zuordnen, die Sie als Hauptbenutzer betrachten.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy— Diese AWS verwaltete Richtlinie bietet Berechtigungen für die Verwendung durch die Rolle, die mit dem AWS Organizations Dienst verknüpft ist. Diese Rolle ermöglicht es Organizations, zusätzliche dienstbezogene Rollen für andere Dienste in Ihrer AWS Organisation zu erstellen.

Allgemeine Warnung – Erstellen Sie SLR mit Stern in Aktion und Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Behebung der allgemeinen Warnung

Die Aktion `iam:CreateServiceLinkedRole` erteilt die Erlaubnis, eine IAM-Rolle zu erstellen, die es einem AWS Dienst ermöglicht, Aktionen in Ihrem Namen auszuführen. Richtlinien mit einem Platzhalter (*) in den Elementen `Action` und `Resource` können die Erstellung unbeabsichtigter serviceverknüpfter Rollen für mehrere Ressourcen ermöglichen. Auf diese Weise können Sie eine dienstbezogene Rolle erstellen, wenn Sie `"Action": "*"` `"Action": "iam:*"`, oder angeben. `"Action": "iam:Create*"` AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben.

- CreateServiceLinkedRole Betrieb
- IAM-JSON-Richtlinienelemente: Aktion
- IAM-JSON-Richtlinienelemente: Resource

AWS verwaltete Richtlinien mit dieser allgemeinen Warnung

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Einige dieser Anwendungsfälle richten sich an Administratoren in Ihrem Konto. Die folgenden AWS verwalteten Richtlinien bieten Administratorzugriff und gewähren Berechtigungen zum Erstellen von dienstbezogenen Rollen für jeden AWS Dienst. AWS empfiehlt, dass Sie die folgenden AWS verwalteten Richtlinien nur den IAM-Identitäten zuordnen, die Sie als Administratoren betrachten.

- AdministratorAccess
- IAM FullAccess

Allgemeine Warnung — Erstellen Sie eine Spiegelreflexkamera mit Stern in der Ressource und NotAction

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Behebung der allgemeinen Warnung

Die Aktion `iam:CreateServiceLinkedRole` erteilt die Erlaubnis, eine IAM-Rolle zu erstellen, die es einem AWS Dienst ermöglicht, Aktionen in Ihrem Namen auszuführen. Die Verwendung des

`NotAction`-Elements in einer Richtlinie mit einem Platzhalter (*) im `Resource`-Element kann dazu führen, dass unbeabsichtigte serviceverknüpfte Rollen für mehrere Ressourcen erstellt werden. AWS empfiehlt, dass Sie stattdessen erlaubte ARNs im `Resource`-Element angeben. Sie können auch `iam:CreateServiceLinkedRole` zu dem Element `NotAction` hinzufügen.

- CreateServiceLinkedRole Betrieb
- Elemente der IAM-JSON-Richtlinie: NotAction
- IAM-JSON-Richtlinienelemente: Aktion
- IAM-JSON-Richtlinienelemente: Resource

Allgemeine Warnung – Veralteter globaler Bedingungsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Behebung der allgemeinen Warnung

Die Richtlinie enthält einen veralteten globalen Bedingungsschlüssel. Aktualisieren Sie den Bedingungsschlüssel im Bedingungsschlüssel-Wert-Paar, um einen unterstützten globalen Bedingungsschlüssel zu verwenden.

- Globale Bedingungsschlüssel

Allgemeine Warnung – Ungültiger Datumswert

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Behebung der allgemeinen Warnung

Die Unix-Epoch-Zeit beschreibt einen Zeitpunkt, der seit dem 1. Januar 1970 verstrichen ist, minus Schaltsekunden. Die Epochenzeit entspricht möglicherweise nicht genau der Zeit, die Sie erwarten. AWS empfiehlt, den W3C-Standard für Datums- und Uhrzeitformate zu verwenden. Sie können zum Beispiel ein vollständiges Datum angeben, wie JJJJ-MM-TT (1997-07-16), oder Sie können auch die Uhrzeit an die Sekunde anfügen, wie JJJJ-MM-TThh:mm:ssTZD (1997-07-16T19:20:30+01:00).

- W3C Datums- und Uhrzeitformate
- IAM JSON-Richtlinienelemente: Version
- aws: CurrentTime globaler Bedingungsschlüssel

Allgemeine Warnung – Ungültige Rollenreferenz

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Behebung der allgemeinen Warnung

AWS empfiehlt, den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle anstelle ihrer Prinzipal-ID anzugeben. Wenn IAM die Richtlinie speichert, wandelt es den ARN in die Prinzipal-ID für die vorhandene Rolle um. AWS beinhaltet eine Sicherheitsmaßnahme. Wenn jemand die Rolle löscht und neu erstellt, hat sie eine neue ID, und die Richtlinie stimmt nicht mit der ID der neuen Rolle überein.

- IAM-Rollen angeben: IAM-Rollen
- IAM-ARNs
- IAM-Eindeutige IDs

Allgemeine Warnung – Ungültige Benutzerreferenz

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Behebung der allgemeinen Warnung

AWS empfiehlt, den Amazon-Ressourcennamen (ARN) für einen IAM-Benutzer anstelle seiner Prinzipal-ID anzugeben. Wenn IAM die Richtlinie speichert, wandelt es den ARN in die Prinzipal-ID für den vorhandenen Benutzer um. AWS beinhaltet eine Sicherheitsmaßnahme. Wenn jemand den Benutzer löscht und neu erstellt, hat er eine neue ID, und die Richtlinie stimmt nicht mit der ID des neuen Benutzers überein.

- IAM Benutzer angeben: IAM-Benutzer
- IAM-ARNs
- IAM-Eindeutige IDs

Allgemeine Warnung – Fehlende Version

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Behebung der allgemeinen Warnung

AWS empfiehlt, dass Sie den optionalen `Version` Parameter in Ihre Richtlinie aufnehmen. Wenn Sie kein Element "Version" einfügen, wird der Wert standardmäßig auf `2012-10-17` gesetzt, aber neuere Features, wie z. B. Richtlinienvariablen, funktionieren nicht mit Ihrer Richtlinie. Beispielsweise werden Variablen wie `${aws:username}` nicht als Variablen erkannt und stattdessen als literale Zeichenketten in der Richtlinie behandelt.

- IAM JSON-Richtlinienelemente: Version

Allgemeine Warnung – Einzigartige Sids empfohlen

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Behebung der allgemeinen Warnung

AWS empfiehlt, eindeutige Anweisungs-IDs zu verwenden. Die `Sid` (Statement-ID) ist eine optionale ID, die Sie in die Richtlinie aufnehmen können. Sie können jeder Anweisung in einem Anweisungsarray einen Anweisungs-ID-Wert zuweisen, indem Sie das `SID`-Element verwenden.

Verwandte Begriffe

- IAM-JSON-Richtlinienelemente: Sid

Allgemeine Warnung – Platzhalter ohne gleicher Operator

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Behebung der allgemeinen Warnung

Die `Condition`-Elementstruktur erfordert, dass Sie einen Bedingungsoperator und ein Schlüssel-Wert-Paar verwenden. Wenn Sie einen Bedingungswert angeben, der einen Platzhalter (*, ?) verwendet, müssen Sie die `Like`-Version des Bedingungsoperators verwenden. Verwenden Sie z. B. anstelle des `StringEquals` Operators für Zeichenkettenbedingungen verwenden Sie `StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- IAM-JSON-Richtlinienelemente: Condition

AWS verwaltete Richtlinien mit dieser allgemeinen Warnung

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Die folgenden AWS verwalteten Richtlinien enthalten Platzhalter in ihrem Bedingungswert ohne einen Bedingungsoperator, der `Like` für den Musterabgleich verwendet wird. Wenn Sie die AWS verwaltete Richtlinie als Referenz für die Erstellung Ihrer vom Kunden verwalteten Richtlinie verwenden, AWS empfiehlt es sich, einen Bedingungsoperator zu verwenden, der den Musterabgleich mit Platzhaltern (*,?) unterstützt , wie zum Beispiel. `StringLike`

- AWSGlueConsoleSageMakerNotebookFullAccess

Allgemeine Warnung – Richtliniengröße überschreitet das Kontingent für Identitätsrichtlinien

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Behebung der allgemeinen Warnung

Sie können einer IAM-Identität (Benutzer, Benutzergruppe oder Rolle) bis zu 10 verwaltete Richtlinien anfügen. Die Größe jeder verwalteten Richtlinie darf jedoch das Standardkontingent von 6.144 Zeichen nicht überschreiten. Leerzeichen werden in IAM beim Berechnen der Richtliniengröße für dieses Kontingent nicht mitgezählt. Kontingente, auch Limits genannt AWS, sind die Höchstwerte für die Ressourcen, Aktionen und Elemente in Ihrem AWS Konto.

Darüber hinaus können Sie einer IAM-Identität beliebig viele Inline-Richtlinien hinzufügen. Die Summe aller Inline-Richtlinien pro Identität darf jedoch das angegebene Kontingent nicht überschreiten.

Wenn Ihre Richtlinie größer als das Kontingent ist, können Sie Ihre Richtlinie in mehreren Anweisungen organisieren und die Anweisungen in mehreren Richtlinien gruppieren.

Verwandte Begriffe

- IAM und AWS STS Charakterkontingente
- Mehrere Anweisungen und mehrere Richtlinien
- Vom IAM-Kunden verwaltete Richtlinien
- Übersicht über JSON-Richtlinien
- IAM JSON-Richtliniengrammatik

AWS verwaltete Richtlinien mit dieser allgemeinen Warnung

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Die folgenden AWS verwalteten Richtlinien gewähren Berechtigungen für Aktionen für viele AWS Dienste und überschreiten die maximale Richtliniengröße. Wenn Sie die verwaltete AWS -Richtlinie als Referenz zum Erstellen Ihrer verwalteten Richtlinie verwenden, müssen Sie die Richtlinie in mehrere Richtlinien aufteilen.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Allgemeine Warnung - Richtliniengröße überschreitet das Kontingent für Ressourcenrichtlinien

Das AWS Management Console Ergebnis dieser Prüfung beinhaltet die folgende Meldung:

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Behebung der allgemeinen Warnung

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource wie einen Amazon-S3-Bucket anfügen. Diese Richtlinien erteilen dem angegebenen Prinzipal die Berechtigung zum Ausführen bestimmter Aktionen für diese Ressource und definiert, unter welchen Bedingungen dies gilt. Die Größe ressourcenbasierter Richtlinien darf das für diese Ressource festgelegte Kontingent nicht überschreiten. Kontingente, auch Limits genannt AWS, sind die Höchstwerte für die Ressourcen, Aktionen und Elemente in Ihrem AWS Konto.

Wenn Ihre Richtlinie größer als das Kontingent ist, können Sie Ihre Richtlinie in mehreren Anweisungen organisieren und die Anweisungen in mehreren Richtlinien gruppieren.

Verwandte Begriffe

- Ressourcenbasierte Richtlinien
- Amazon S3-Bucket-Richtlinien

- Mehrere Anweisungen und mehrere Richtlinien
- Übersicht über JSON-Richtlinien
- IAM JSON-Richtliniengrammatik

Allgemeine Warnung – Typenabweichung

Das AWS Management Console Ergebnis dieser Prüfung enthält die folgende Meldung:

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Behebung der allgemeinen Warnung

Aktualisieren Sie den Text, um den unterstützten Bedingungsoperator Datentyp zu verwenden.

Zum Beispiel erfordert der globale Konditionsschlüssel `aws:MultiFactorAuthPresent` einen Konditionsoperator mit dem Datentyp `Boolean`. Wenn Sie ein Datum oder eine Ganzzahl angeben, stimmt der Datentyp nicht überein.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Allgemeine Warnung – Typkonflikt Boolescher Wert

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Behebung der allgemeinen Warnung

Aktualisieren Sie den Text, um einen booleschen Bedingungsoperator als Datentyp zu verwenden, z. B. `true` oder `false`.

Zum Beispiel erfordert der globale Konditionsschlüssel `aws:MultiFactorAuthPresent` einen Konditionsoperator mit dem Datentyp `Boolean`. Wenn Sie ein Datum oder eine Ganzzahl angeben, stimmt der Datentyp nicht überein.

Verwandte Begriffe

- Boolesche Bedingungsoperatoren
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Allgemeine Warnung – Datum des Typs, das nicht übereinstimmt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Behebung der allgemeinen Warnung

Aktualisieren Sie den Text so, dass er den Datentyp "Datumsbedingungsoperator" im Format `YYYY-MM-DD` oder einem anderen ISO-8601-Datumszeitformat verwendet.

Verwandte Begriffe

- Bedingungsoperatoren für Datum
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Allgemeine Warnung – Typ-Unübereinstimmungsnummer

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Behebung der allgemeinen Warnung

Aktualisieren Sie den Text, um den numerischen Bedingungsoperator Datentyp zu verwenden.

Verwandte Begriffe

- Numerische Bedingungsoperatoren
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Allgemeine Warnung – Zeichenfolge, die nicht übereinstimmt

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Behebung der allgemeinen Warnung

Aktualisieren Sie den Text, um den Datentyp „Zeichenfolgenbedingung“ zu verwenden.

Verwandte Begriffe

- Bedingungsoperatoren für Zeichenfolgen
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Allgemeine Warnung – Spezifische(s) Github-Repository und -Branch empfohlen

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value of
 token.actions.githubusercontent.com:sub with the repository and branch name."
```

Behebung der allgemeinen Warnung

Wenn Sie GitHub als OIDC-IdP verwenden, empfiehlt es sich, die Entitäten einzuschränken, die die dem IAM-IdP zugeordnete Rolle übernehmen können. Wenn Sie eine `Condition` Erklärung in eine Vertrauensrichtlinie für Rollen aufnehmen, können Sie die Rolle auf eine bestimmte GitHub Organisation, ein Repository oder eine Branche beschränken. Sie können den Bedingungsschlüssel `token.actions.githubusercontent.com:sub` verwenden, um den Zugriff einzuschränken. Wir empfehlen, dass Sie die Bedingung auf eine bestimmte Gruppe von Repositorys oder Branchen beschränken. Wenn Sie einen Platzhalter (`*`) in verwenden`token.actions.githubusercontent.com:sub`, können GitHub Aktionen von Organisationen oder Repositorys, auf die Sie keinen Einfluss haben, Rollen übernehmen, die dem GitHub IAM-IdP in Ihrem Konto zugeordnet sind. AWS

Verwandte Begriffe

- Konfiguration einer Rolle für den OIDC-Identitätsanbieter GitHub

Allgemeine Warnung – Richtliniengröße überschreitet das Kontingent für Rollenvertrauensrichtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Behebung der allgemeinen Warnung

IAM und AWS STS verfügen über Kontingente, die den Umfang der Richtlinien für die Vertrauensstellung von Rollen einschränken. Die Zeichen in der Rollenvertrauensrichtlinie, ausgenommen Leerzeichen, überschreiten das Zeichenmaximum. Wir empfehlen, dass Sie eine Kontingenterhöhung für die Länge der Rollenvertrauensrichtlinie anfordern, indem Sie Service Quotas und die AWS Support Center Console verwenden.

Verwandte Begriffe

- IAM und AWS STS Kontingente, Namensanforderungen und Zeichenbeschränkungen

Sicherheitswarnung — Zulassen mit NotPrincipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Auflösen der Sicherheitswarnung

Die Verwendung von `"Effect": "Allow"` mit der `NotPrincipal` kann zu permissiv sein. Dies kann beispielsweise anonymen Prinzipalen Berechtigungen gewähren. AWS empfiehlt, dass Sie mithilfe des Elements Prinzipale angeben, die `Principal` Zugriff benötigen. Alternativ können Sie den breiten Zugang erlauben und dann eine weitere Anweisung hinzufügen, die das Element `NotPrincipal` mit `“Effect”: “Deny”` verwendet.

- AWS JSON-Richtlinienelemente: Prinzipal
- AWS JSON-Richtlinienelemente: NotPrincipal

Sicherheitswarnung — ForAllValues mit einwertigem Schlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Auflösen der Sicherheitswarnung

AWS empfiehlt, die Option `ForAllValues` nur mit mehrwertigen Bedingungen zu verwenden. Der `ForAllValues`-Set-Operator prüft, ob der Wert eines jeden Mitglieds der Anforderungsmenge eine Teilmenge der Bedingungsschlüsselmenge ist. Die Bedingung gibt "true" zurück, wenn jeder Schlüsselwert in der Anforderung mindestens einem Wert in der Richtlinie entspricht. „true“ wird zudem zurückgegeben, wenn keine Schlüssel in der Anforderung vorhanden sind oder wenn die Schlüsselwerte zu einem Null-Dataset aufgelöst werden, z. B. einer leeren Zeichenfolge.

Um zu erfahren, ob eine Bedingung einen einzelnen Wert oder mehrere Werte unterstützt, lesen Sie die Seite Aktionen, Ressourcen und Bedingungsschlüssel für den Service. Bedingungsschlüssel mit`ArrayOf`Datentyppräfix sind Mehrfachwertbedingungsschlüssel. Amazon SES unterstützt beispielsweise Schlüssel mit Einzelwerten (`String`) und den `ArrayOfString`-Mehrwertigen Datentyp.

- Mehrwertige Kontextschlüssel

Sicherheitswarnung — Rolle übergeben mit NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Auflösen der Sicherheitswarnung

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Um dies zu ermöglichen, müssen Sie einer Identität (Benutzer, Benutzergruppe oder Rolle) die Berechtigung `iam:PassRole` erteilen. Wenn Sie das `NotResource` Element `iam:PassRole` in einer Richtlinie verwenden, können Ihre Prinzipale möglicherweise auf mehr Dienste oder Funktionen zugreifen, als Sie beabsichtigt haben. AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben. Darüber hinaus können Sie Berechtigungen auf ein einzelnes Service reduzieren, indem Sie die `iam:PassedToService`-Bedingungsschlüssel verwenden.

- Übergehen einer Rolle an einen Service
- ich bin: PassedToService
- Elemente der IAM-JSON-Richtlinie: NotResource
- IAM-JSON-Richtlinienelemente: Resource

Sicherheitswarnung — Übergeben Sie die Rolle mit Star in Aktion und NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Auflösen der Sicherheitswarnung

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Um dies zu ermöglichen, müssen Sie einer Identität (Benutzer, Benutzergruppe oder Rolle) die Berechtigung `iam:PassRole` erteilen. Richtlinien, die einen Platzhalter (*) enthalten `Action` und das `NotResource` Element enthalten, können es Ihren Prinzipalen ermöglichen, auf mehr Dienste oder Funktionen zuzugreifen, als Sie beabsichtigt haben. AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben. Darüber hinaus können Sie Berechtigungen auf ein einzelnes Service reduzieren, indem Sie die `iam:PassedToService`-Bedingungsschlüssel verwenden.

- Übergehen einer Rolle an einen Service
- ich bin: PassedToService
- Elemente der IAM-JSON-Richtlinie: NotResource
- IAM-JSON-Richtlinienelemente: Resource

Sicherheitswarnung — Rolle mit NotAction und übergeben NotResource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Auflösen der Sicherheitswarnung

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Um dies zu ermöglichen, müssen Sie einer Identität (Benutzer, Benutzergruppe oder Rolle) die Berechtigung `iam:PassRole` erteilen. Wenn Sie das `NotAction` Element verwenden und einige Ressourcen im `NotResource` Element auflisten, können Ihre Prinzipale auf mehr Dienste oder Funktionen zugreifen, als Sie beabsichtigt haben. AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben. Darüber hinaus können Sie Berechtigungen auf ein einzelnes Service reduzieren, indem Sie die `iam:PassedToService`-Bedingungsschlüssel verwenden.

- Übergehen einer Rolle an einen Service
- ich bin: PassedToService
- Elemente der IAM-JSON-Richtlinie: NotAction
- IAM-JSON-Richtlinienelemente: Aktion
- IAM-JSON-Richtlinienelemente: NotResource
- IAM-JSON-Richtlinienelemente: Resource

Sicherheitswarnung – Übergeben der Rolle mit Stern in der Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Auflösen der Sicherheitswarnung

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Um dies zu ermöglichen, müssen Sie einer Identität (Benutzer, Benutzergruppe oder Rolle) die Berechtigung `iam:PassRole` erteilen. Richtlinien, die das `Resource` Element zulassen `iam:PassRole` und

einen Platzhalter (*) enthalten, können es Ihren Prinzipalen ermöglichen, auf mehr Dienste oder Funktionen zuzugreifen, als Sie beabsichtigt haben. AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben. Darüber hinaus können Sie Berechtigungen auf ein einzelnes Service reduzieren, indem Sie die `iam:PassedToService`-Bedingungsschlüssel verwenden.

Einige AWS Dienste schließen ihren Dienst-Namespace im Namen ihrer Rolle ein. Bei dieser Richtlinienprüfung werden diese Konventionen bei der Analyse der Richtlinie zum Generieren von Ergebnissen berücksichtigt. Beispiel: Die folgende Ressourcen-ARN generiert möglicherweise keine Ergebnisse:

```
arn:aws:iam::*:role/Service*
```

- Übergehen einer Rolle an einen Service
- ich bin: PassedToService
- IAM-JSON-Richtlinienelemente: Resource

AWS verwaltete Richtlinien mit dieser Sicherheitswarnung

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Einer dieser Anwendungsfälle gilt für Administratoren in Ihrem Konto. Die folgenden AWS verwalteten Richtlinien bieten Administratorzugriff und gewähren Berechtigungen zur Weitergabe beliebiger IAM-Rollen an einen beliebigen Dienst. AWS empfiehlt, die folgenden AWS verwalteten Richtlinien nur IAM-Identitäten zuzuordnen, die Sie als Administratoren betrachten.

- AdministratorAccess-Verstärken

Die folgenden AWS verwalteten Richtlinien enthalten Berechtigungen für die Verwendung `iam:PassRole` eines Platzhalters (*) in der Ressource und befinden sich in einem veralteten Pfad. Für jede dieser Richtlinien haben wir die Richtlinien zu Berechtigungen aktualisiert und beispielsweise eine neue AWS verwaltete Richtlinie empfohlen, die diesen Anwendungsfall unterstützt. Alternativen zu diesen Richtlinien finden Sie in den Leitfäden der einzelnen Services.

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess

- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDB FullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDB FullAccesswithDataPipeline
- Amazon EC2 ContainerServiceFullAccess

Die folgenden AWS verwalteten Richtlinien gewähren nur Berechtigungen für dienstbezogene Rollen, sodass AWS Dienste Aktionen in Ihrem Namen ausführen können. Sie können diese Richtlinie an Ihre IAM-Identitäten anfügen.

- AWSServiceRoleForAmazonEKSNodegroup

Sicherheitswarnung – Übergeben Sie die Rolle mit Stern in Aktion und Ressource

Das AWS Management Console Ergebnis dieser Prüfung enthält die folgende Meldung:

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Auflösen der Sicherheitswarnung

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Um dies zu ermöglichen, müssen Sie einer Identität (Benutzer, Benutzergruppe oder Rolle) die Berechtigung `iam:PassRole` erteilen. Richtlinien mit einem Platzhalter (*) in den `Resource` Elementen `Action`

und können es Ihren Prinzipalen ermöglichen, auf mehr Dienste oder Funktionen zuzugreifen, als Sie beabsichtigt haben. AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben. Darüber hinaus können Sie Berechtigungen auf ein einzelnes Service reduzieren, indem Sie die `iam:PassedToService`-Bedingungsschlüssel verwenden.

- Übergehen einer Rolle an einen Service
- ich bin: PassedToService
- IAM-JSON-Richtlinienelemente: Aktion
- IAM-JSON-Richtlinienelemente: Resource

AWS verwaltete Richtlinien mit dieser Sicherheitswarnung

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Einige dieser Anwendungsfälle richten sich an Administratoren in Ihrem Konto. Die folgenden AWS verwalteten Richtlinien bieten Administratorzugriff und gewähren Berechtigungen zur Weitergabe beliebiger IAM-Rollen an einen beliebigen AWS Dienst. AWS empfiehlt, dass Sie die folgenden AWS verwalteten Richtlinien nur den IAM-Identitäten zuordnen, die Sie als Administratoren betrachten.

- AdministratorAccess
- IAM FullAccess

Sicherheitswarnung — Übergeben Sie die Rolle mit Stern in den Kategorien Ressource und NotAction

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
```

```
that you specify resource ARNs or add the iam:PassedToService condition key to your
statement."
```

Auflösen der Sicherheitswarnung

Um viele AWS Dienste zu konfigurieren, müssen Sie dem Dienst eine IAM-Rolle übergeben. Um dies zu ermöglichen, müssen Sie einer Identität (Benutzer, Benutzergruppe oder Rolle) die Berechtigung `iam:PassRole` erteilen. Wenn Sie das `NotAction` Element in einer Richtlinie mit einem Platzhalter (*) im `Resource` Element verwenden, können Ihre Prinzipale auf mehr Dienste oder Funktionen zugreifen, als Sie beabsichtigt haben. AWS empfiehlt, stattdessen zulässige ARNs im `Resource` Element anzugeben. Darüber hinaus können Sie Berechtigungen auf ein einzelnes Service reduzieren, indem Sie die `iam:PassedToService`-Bedingungsschlüssel verwenden.

- Übergehen einer Rolle an einen Service
- ich bin: PassedToService
- Elemente der IAM-JSON-Richtlinie: NotAction
- IAM-JSON-Richtlinienelemente: Aktion
- IAM-JSON-Richtlinienelemente: Resource

Sicherheitswarnung – Fehlende gekoppelte Zustandsschlüssel

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Auflösen der Sicherheitswarnung

Einige Bedingungsschlüssel sind sicherer, wenn sie mit anderen zugehörigen Zustandstasten gekoppelt werden. AWS empfiehlt, dass Sie die zugehörigen Bedingungsschlüssel in denselben Bedingungsblock wie der vorhandene Bedingungsschlüssel aufnehmen. Dadurch werden die durch die Richtlinie gewährten Berechtigungen sicherer.

Sie können zum Beispiel den `aws:VpcSourceIp` Bedingungsschlüssel verwenden, um die IP-Adresse, von der eine Anforderung gestellt wurde, mit der IP-Adresse zu vergleichen, die Sie in der Richtlinie angeben. AWS empfiehlt, dass Sie den entsprechenden `aws:SourceVPC` Bedingungsschlüssel hinzufügen. Dabei wird geprüft, ob die Anforderung von der in der Richtlinie angegebenen VPC und der angegebenen IP-Adresse stammt.

Verwandte Begriffe

- `aws:VpcSourceIp` (globaler -Bedingungsschlüssel)
- `aws:SourceVPC` (globaler -Bedingungsschlüssel)
- Globale Bedingungsschlüssel
- Condition-Element
- Übersicht über JSON-Richtlinien

Sicherheitswarnung – Verweigern mit nicht unterstütztem Tag-Bedingungsschlüssel für das Service

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Auflösen der Sicherheitswarnung

Die Verwendung von nicht unterstützten Tag-Bedingungsschlüsseln im `Condition` Element einer Richtlinie mit `"Effect": "Deny"` kann zu freizügig sein, da die Bedingung für diesen Dienst ignoriert wird. AWS empfiehlt, die Dienstaktionen zu entfernen, die den Bedingungsschlüssel nicht unterstützen, und eine weitere Anweisung zu erstellen, um den Zugriff auf bestimmte Ressourcen für diese Aktionen zu verweigern.

Wenn Sie den `aws:ResourceTag`-Bedingungsschlüssel verwenden und dieser nicht von einer Service-Aktion unterstützt wird, wird der Schlüssel nicht in den Anforderungskontext aufgenommen. In diesem Fall gibt die Bedingung in der Anweisung `Deny` immer `false` zurück und die Aktion wird nie verweigert. Dies geschieht auch dann, wenn die Ressource korrekt markiert ist.

Wenn ein Service den Bedingungsschlüssel `aws:ResourceTag` unterstützt, können Sie den Zugriff auf die Ressourcen dieses Services mit Hilfe von Tags steuern. Dies wird als attributbasierte Zugriffskontrolle (ABAC) bezeichnet. Bei Services, die diese Schlüssel nicht unterstützen, müssen Sie den Zugriff auf Ressourcen mithilfe der ressourcenbasierten Zugriffskontrolle (RBAC) steuern.

> **Note**
>
> Einige Services ermöglichen die Unterstützung des Bedingungsschlüssel `aws:ResourceTag` für eine Teilmenge ihrer Ressourcen und Aktionen. IAM Access Analyzer gibt Ergebnisse für die Service-Aktionen zurück, die nicht unterstützt werden. Amazon S3 unterstützt beispielsweise `aws:ResourceTag` für eine Teilmenge der Ressourcen. Alle in Amazon S3 verfügbaren Ressourcentypen, die den `aws:ResourceTag`-Bedingungsschlüssel unterstützen, finden Sie unter Resource types defined by Amazon S3 in der Service Authorization Reference.

Nehmen Sie beispielsweise an, dass Sie den Zugriff auf bestimmte Ressourcen, die mit dem Schlüssel-Wert-Paar `status=Confidential` gekennzeichnet sind, verweigern möchten. Gehen Sie außerdem davon aus, dass Sie damit Ressourcen kennzeichnen und die Markierung aufheben AWS Lambda können, der `aws:ResourceTag` Bedingungsschlüssel jedoch nicht unterstützt wird. Verwenden Sie den `aws:ResourceTag` Bedingungsschlüssel, um die Löschaktionen für AWS App Mesh und AWS Backup falls dieses Tag vorhanden ist, zu verweigern. Verwenden Sie für Lambda eine Ressourcennamenskonvention, die das Präfix `"Confidential"` enthält. Fügen Sie dann eine separate Anweisung hinzu, die das Löschen von Ressourcen mit dieser Namenskonvention verhindert.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> Verwenden Sie nicht die IfExistsVersion... des Bedingungsoperators, um dieses Problem zu umgehen. Dies bedeutet „Die Aktion verweigern, wenn der Schlüssel im Anforderungskontext vorhanden ist und die Werte übereinstimmen. Andernfalls verweigern Sie die Aktion.“ Im vorangegangenen Beispiel wird durch die Aufnahme der Aktion `lambda:DeleteFunction` in die Anweisung `DenyDeleteSupported` mit dem Operator `StringEqualsIfExists` die Aktion immer verweigert. Für diese Aktion ist der Schlüssel nicht im Kontext vorhanden, und jeder Versuch, diesen Ressourcentyp zu löschen, wird verweigert, unabhängig davon, ob die Ressource mit Tags versehen ist.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- ABAC mit RBAC vergleichen

- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- Condition-Element
- Übersicht über JSON-Richtlinien

Sicherheitswarnung — NotAction Mit nicht unterstütztem Tag-Bedingungsschlüssel für den Dienst ablehnen

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Auflösen der Sicherheitswarnung

Die Verwendung von Tag-Bedingungsschlüsseln im Element `Condition` einer Richtlinie mit den Elementen `NotAction` und `"Effect": "Deny"` kann übermäßig permissiv sein. Die Bedingung wird für Serviceaktionen ignoriert, die den Bedingungsschlüssel nicht unterstützen. AWS empfiehlt, die Logik so umzuschreiben, dass eine Liste von Aktionen verweigert wird.

Wenn Sie den Bedingungsschlüssel `aws:ResourceTag` mit `NotAction` verwenden, werden alle neuen oder bestehenden Service-Aktionen, die den Schlüssel nicht unterstützen, nicht verweigert. AWS empfiehlt, die Aktionen, die verweigert werden sollen, explizit aufzulisten. IAM Access Analyzer gibt eine separate Suche für aufgelistete Aktionen zurück, die die `aws:ResourceTag`-Bedingungsschlüssel. Weitere Informationen finden Sie unter Sicherheitswarnung – Verweigern mit nicht unterstütztem Tag-Bedingungsschlüssel für das Service.

Wenn ein Service den Bedingungsschlüssel `aws:ResourceTag` unterstützt, können Sie den Zugriff auf die Ressourcen dieses Services mit Hilfe von Tags steuern. Dies wird als attributbasierte

Zugriffskontrolle (ABAC) bezeichnet. Bei Services, die diese Schlüssel nicht unterstützen, müssen Sie den Zugriff auf Ressourcen mithilfe der ressourcenbasierten Zugriffskontrolle (RBAC) steuern.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- ABAC mit RBAC vergleichen
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- Condition-Element
- Übersicht über JSON-Richtlinien

Sicherheitswarnung - Zugriff auf Service-Prinzipal einschränken

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Auflösen der Sicherheitswarnung

Sie können AWS-Services im `Principal` Element einer ressourcenbasierten Richtlinie einen Dienstprinzipal angeben, bei dem es sich um einen Bezeichner für den Dienst handelt. Wenn Sie Zugang zu einem Service-Prinzipal gewähren, um in Ihrem Namen zu handeln, beschränken Sie den Zugriff. Sie können übermäßig freizügige Richtlinien verhindern, indem Sie die `aws:SourceOrgPaths` Bedingungsschlüssel `aws:SourceArnaws:SourceAccount`,`aws:SourceOrgID`, oder verwenden, um den Zugriff auf eine bestimmte Quelle zu beschränken, z. B. einen bestimmten Ressourcen-ARN AWS-Konto, eine Organisations-ID oder Organisationspfade. Durch die Einschränkung des Zugriffs können Sie ein Sicherheitsproblem verhindern, das Problem der verwirrten Stellvertreters genannt wird.

Verwandte Begriffe

- AWS-Service Prinzipale
- AWS globale Bedingungsschlüssel: aws: SourceAccount
- AWS globale Bedingungsschlüssel: aws: SourceArn
- AWS globale Bedingungsschlüssel: aws: SourceOrgId
- AWS globale Bedingungsschlüssel: aws: SourceOrgPaths
- Das Confused-Deputy-Problem

Sicherheitswarnung – Fehlender Bedingungsschlüssel für OIDC-Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Auflösen der Sicherheitswarnung

Die Verwendung eines Open ID Connect-Prinzipals ohne Bedingung kann zu großzügig sein. Fügen Sie Bedingungsschlüssel mit einem Präfix hinzu, das Ihren OIDC-Verbundprinzipalen entspricht, um sicherzustellen, dass nur der beabsichtigte Identitätsanbieter die Rolle übernimmt.

Verwandte Begriffe

- Erstellen einer Rolle für Web-Identität oder OpenID-Connect-Verbund (Konsole)

Sicherheitswarnung – Fehlender Bedingungsschlüssel für Github-Repository

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
```

```
Add the token.actions.githubusercontent.com:sub condition key and specify the branch
and repository name in the value.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

Auflösen der Sicherheitswarnung

Wenn Sie GitHub als OIDC-IdP verwenden, empfiehlt es sich, die Entitäten einzuschränken, die die dem IAM-IdP zugeordnete Rolle übernehmen können. Wenn Sie eine `Condition` Erklärung in eine Vertrauensrichtlinie für Rollen aufnehmen, können Sie die Rolle auf eine bestimmte GitHub Organisation, ein Repository oder eine Branche beschränken. Sie können den Bedingungsschlüssel `token.actions.githubusercontent.com:sub` verwenden, um den Zugriff einzuschränken. Wir empfehlen, dass Sie die Bedingung auf eine bestimmte Gruppe von Repositorys oder Branchen beschränken. Wenn Sie diese Bedingung nicht angeben, können GitHub Aktionen von Organisationen oder Repositorys, auf die Sie keinen Einfluss haben, Rollen übernehmen, die dem GitHub IAM-IdP in Ihrem Konto zugeordnet sind. AWS

Verwandte Begriffe

- Konfiguration einer Rolle für GitHub den OIDC-Identitätsanbieter

Vorschlag – Leere Array-Aktion

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Abruf eines Vorschlags

Aussagen müssen entweder ein `Action` oder `NotAction` Element enthalten, das eine Reihe von Aktionen umfasst. Wenn das Element leer ist, stellt die Richtlinienanweisung keine Berechtigungen bereit. Geben Sie Aktionen im Element `Action` an.

- IAM-JSON-Richtlinienelemente: Aktion

Vorschlag – Leere Array-Bedingung

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Abruf eines Vorschlags

Die optionale `Condition`-Element-Struktur erfordert die Verwendung eines Bedingungsoperators und eines Schlüssel-Wert-Paares. Wenn der Wert der Bedingung leer ist, gibt die Bedingung `true` zurück und die Richtlinienanweisung sieht keine Berechtigungen vor. Geben Sie einen Bedingungswert an.

- IAM-JSON-Richtlinienelemente: Condition

Vorschlag — Zustand „Leeres Array“ ForAllValues

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Abruf eines Vorschlags

Die `Condition`-Elementstruktur erfordert, dass Sie einen Bedingungsoperator und ein Schlüssel-Wert-Paar verwenden. Der `ForAllValues`-Set-Operator prüft, ob der Wert eines jeden Mitglieds der Anforderungsmenge eine Teilmenge der Bedingungsschlüsselmenge ist.

Wenn Sie `ForAllValues` mit einem leeren Bedingungsschlüssel verwenden, trifft die Bedingung nur dann zu, wenn es keine Schlüssel in der Anforderung gibt. AWS empfiehlt, dass Sie stattdessen den Bedingungsoperator verwenden, wenn Sie `Null` testen wollen, ob ein Anforderungskontext leer ist.

- Mehrwertige Kontextschlüssel
- Null-Bedingungs-Operator
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag — Zustand „Leeres Array“ ForAnyValue

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Abruf eines Vorschlags

Die `Condition`-Elementstruktur erfordert, dass Sie einen Bedingungsoperator und ein Schlüssel-Wert-Paar verwenden. Der `ForAnyValues`-Set-Operator prüft, ob mindestens ein

Mitglied der Menge der Anforderungswerte mit mindestens einem Mitglied der Menge der Bedingungsschlüsselwerte übereinstimmt.

Wenn Sie `ForAnyValues` mit einem leeren Bedingungsschlüssel verwenden, wird die Bedingung nie erfüllt. Das bedeutet, dass die Aussage keine Auswirkung auf die Richtlinie hat. AWS empfiehlt, die Bedingung neu zu schreiben.

- Mehrwertige Kontextschlüssel
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag — Bedingung für leeres Array IfExists

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Abruf eines Vorschlags

Mit dem Suffix `...IfExists` wird ein Bedingungsoperator bearbeitet. Das bedeutet, dass, wenn der Richtlinienschlüssel im Kontext der Anforderung vorhanden ist, der Schlüssel wie in der Richtlinie angegeben verarbeitet wird. Wenn der Schlüssel nicht vorhanden ist, wird das Bedingungselement als "true" ausgewertet.

Wenn Sie `...IfExists` mit einem leeren Bedingungsschlüssel verwenden, trifft die Bedingung nur dann zu, wenn es keine Schlüssel in der Anforderung gibt. AWS empfiehlt, dass Sie stattdessen den Bedingungsoperator verwenden, wenn Sie `Null` testen wollen, ob ein Anforderungskontext leer ist.

- ... IfExists Bedingungsoperatoren
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag – Leerer Array-Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Abruf eines Vorschlags

Sie müssen das Element `Principal` oder `NotPrincipal` in den Vertrauensrichtlinien für IAM-Rollen und in ressourcenbasierten Richtlinien verwenden. Ressourcenbasierte Richtlinien sind Richtlinien, die Sie direkt in eine Ressource einbinden.

Wenn Sie im `Principal` Element einer Anweisung ein leeres Array angeben, hat die Anweisung keine Auswirkung auf die Richtlinie. AWS empfiehlt, dass Sie die Prinzipale angeben, die Zugriff auf die Ressource haben sollen.

- IAM JSON-Richtlinienelemente: Prinzipal
- IAM-JSON-Richtlinienelemente: NotPrincipal

Vorschlag – Leere Array-Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Abruf eines Vorschlags

Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten.

Wenn Sie im Ressourcenelement einer Anweisung ein leeres Array angeben, hat die Anweisung keine Auswirkung auf die Richtlinie. AWS empfiehlt, Amazon Resource Names (ARNs) für Ressourcen anzugeben.

- IAM-JSON-Richtlinienelemente: Resource
- IAM-JSON-Richtlinienelemente: NotResource

Vorschlag – Leere Objektbedingung

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Abruf eines Vorschlags

Die `Condition`-Elementstruktur erfordert, dass Sie einen Bedingungsoperator und ein Schlüssel-Wert-Paar verwenden.

Wenn Sie ein leeres Objekt im Bedingungselement einer Anweisung angeben, hat die Anweisung keine Auswirkung auf die Richtlinie. Entfernen Sie das optionale Element, oder geben Sie Bedingungen an.

- IAM-JSON-Richtlinienelemente: Condition

Vorschlag – Leerer Objekt-Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Abruf eines Vorschlags

Sie müssen das Element `Principal` oder `NotPrincipal` in den Vertrauensrichtlinien für IAM-Rollen und in ressourcenbasierten Richtlinien verwenden. Ressourcenbasierte Richtlinien sind Richtlinien, die Sie direkt in eine Ressource einbinden.

Wenn Sie im `Principal` Element einer Anweisung ein leeres Objekt angeben, hat die Anweisung keine Auswirkung auf die Richtlinie. AWS empfiehlt, dass Sie die Prinzipale angeben, die Zugriff auf die Ressource haben sollen.

- IAM JSON-Richtlinienelemente: Prinzipal
- IAM-JSON-Richtlinienelemente: NotPrincipal

Vorschlag – Leerer Sid Wert

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Abruf eines Vorschlags

Mit dem optionalen Element `Sid` (Ausweis-ID) können Sie einen Identifikator eingeben, den Sie für den Ausweis bereitstellen. Sie können jeder Anweisung in einem Statement-Array einen `Sid`-Wert zuweisen. Wenn Sie das Element `Sid` verwenden, müssen Sie einen String-Wert angeben.

Verwandte Begriffe

- IAM-JSON-Richtlinienelemente: Sid

Vorschlag – Verbessern des IP-Bereichs

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Abruf eines Vorschlags

Die IP-Adressbedingungen müssen im Standard-CIDR-Format vorliegen, z. B. 203.0.113.0/24 oder 2001:DB8:1234:5678::/64. Wenn Sie nach den maskierten Bits Bits ungleich Null angeben, werden sie für die Bedingung nicht berücksichtigt. AWS empfiehlt, dass Sie die neue Adresse verwenden, die in der Nachricht enthalten ist.

- Bedingungsoperatoren für IP-Adressen
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag – Null mit Qualifier

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Abruf eines Vorschlags

Im `Condition`-Element formulieren Sie Ausdrücke, in denen Sie Bedingungsoperatoren verwenden (gleich, kleiner als usw.), um die Bedingungsschlüssel und -werte der Richtlinie mit den Schlüsseln

und Werten in der Anforderungen abzugleichen. Für Anforderungen, die mehrere Werte für einen einzigen Konditionsschlüssel enthalten, müssen Sie die `ForAllValues` oder `ForAnyValue` Set-Operatoren verwenden.

Wenn Sie den Bedingungsoperator `Null` mit `ForAllValues` verwenden, gibt die Anweisung immer `true` zurück. Wenn Sie den `Null` Bedingungsoperator with verwenden`ForAnyValue`, kehrt die Anweisung immer zurück`false`. AWS empfiehlt, den `StringLike` Bedingungsoperator mit diesen Mengenoperatoren zu verwenden.

Verwandte Begriffe

- Mehrwertige Kontextschlüssel
- Null-Bedingungs-Operator
- Condition-Element

Private Empfehlung – Private IP-Adresse

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Abruf eines Vorschlags

Der globale Bedingungsschlüssel `aws:SourceIp` funktioniert nur für öffentliche IP-Adressbereiche.

Wenn Ihre `Condition`-Element eine Mischung aus privaten und öffentlichen IP-Adressen enthält, hat die Anweisung möglicherweise nicht die gewünschte Wirkung. Sie können private IP-Adressen mit `aws:VpcSourceIP` angeben.

> **Note**
>
> Der globale Bedingungsschlüssel `aws:VpcSourceIP` trifft nur zu, wenn die Anforderung von der angegebenen IP-Adresse ausgeht und über einen VPC-Endpunkt läuft.

- aws: SourceIp globaler Bedingungsschlüssel
- aws: VpcSourceIp globaler Bedingungsschlüssel
- Bedingungsoperatoren für IP-Adressen
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag — Private NotIpAddress Teilmenge

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Abruf eines Vorschlags

Der globale Bedingungsschlüssel `aws:SourceIp` funktioniert nur für öffentliche IP-Adressbereiche.

Wenn Ihre `Condition`-Element enthält die `NotIpAddress`-Bedingungsoperator und eine Mischung aus privaten und öffentlichen IP-Adressen, hat die Anweisung möglicherweise nicht den gewünschten Effekt. Alle öffentlichen IP-Adressen, die nicht in der Richtlinie angegeben sind, stimmen überein. Es werden keine privaten IP-Adressen übereinstimmen. Um diesen Effekt zu erzielen, können Sie `NotIpAddress` mit `aws:VpcSourceIP` verwenden und die privaten IP-Adressen angeben, die nicht übereinstimmen sollen.

- aws: SourceIp globaler Bedingungsschlüssel
- aws: VpcSourceIp globaler Bedingungsschlüssel
- Bedingungsoperatoren für IP-Adressen
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag – Redundante Aktion

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Abruf eines Vorschlags

Wenn Sie Platzhalter (*) im `Action` Element verwenden, können Sie redundante Berechtigungen hinzufügen. AWS empfiehlt, dass Sie Ihre Richtlinie überprüfen und nur die Berechtigungen angeben, die Sie benötigen. Auf diese Weise können Sie redundante Aktionen entfernen.

Die folgenden Aktionen umfassen beispielsweise die `iam:GetCredentialReport` zweimal Action (Aktion).

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

In diesem Beispiel werden die Berechtigungen für jede IAM-Aktion definiert, die mit `Get` oder `List` beginnt. Wenn IAM zusätzliche Abruf- oder Listenvorgänge hinzufügt, erlaubt diese Richtlinie diese. Sie können alle diese schreibgeschützten Aktionen zulassen. Die Aktion

`iam:GetCredentialReport` ist bereits als Teil von `iam:Get*` enthalten. Um die doppelten Berechtigungen zu entfernen, könnten Sie `iam:GetCredentialReport` entfernen.

Sie erhalten ein Ergebnis für diese Richtlinienprüfung, wenn der gesamte Inhalt einer Aktion redundant ist. Wenn das Element in diesem Beispiel `iam:*CredentialReport` enthält, wird es nicht als redundant betrachtet. Das schließt mit ein`iam:GetCredentialReport`, die redundant ist, und `iam:GenerateCredentialReport`, was nicht ist. Das Entfernen von `iam:Get*` oder `iam:*CredentialReport` würde die Berechtigungen der Richtlinie ändern.

- IAM-JSON-Richtlinienelemente: Aktion

AWS verwaltete Richtlinien mit diesem Vorschlag

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Redundante Aktionen wirken sich nicht auf die Berechtigungen aus, die von der Richtlinie gewährt werden. Wenn Sie eine AWS verwaltete Richtlinie als Referenz für die Erstellung Ihrer vom Kunden verwalteten Richtlinie verwenden, AWS empfiehlt Ihnen, überflüssige Aktionen aus Ihrer Richtlinie zu entfernen.

Vorschlag – Redundanter Bedingungswert num

Das AWS Management Console Ergebnis dieser Prüfung enthält die folgende Meldung:

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Abruf eines Vorschlags

Wenn Sie numerische Bedingungsoperatoren für ähnliche Werte in einem Bedingungsschlüssel verwenden, können Sie eine Überlappung erstellen, die zu redundanten Berechtigungen führt.

Das folgende `Condition`-Element enthält beispielsweise mehrere `aws:MultiFactorAuthAge`-Bedingungen mit einer Altersüberschneidung von 1200 Sekunden.

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
            "2700",
            "3600"
          ]
        }
      }
```

In diesem Beispiel werden die Berechtigungen definiert, wenn die Multi-Faktor-Authentifizierung (MFA) vor weniger als 3600 Sekunden (1 Stunde) abgeschlossen wurde. Sie könnten die redundante `2700`-Wert.

- Numerische Bedingungsoperatoren
- IAM-JSON-Richtlinienelemente: Condition

Vorschlag – Redundante Ressource

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Abruf eines Vorschlags

Wenn Sie Platzhalter (*) in Amazon-Ressourcennamen (ARNs) verwenden, können Sie redundante Ressourcenberechtigungen erstellen.

So sehen beispielsweise die folgenden `Resource`-Element enthält mehrere ARNs mit redundanten Berechtigungen.

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
```

```
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

In diesem Beispiel sind die Berechtigungen für jede Rolle definiert, deren Name mit `jane`. Sie können die redundanten ARNs `jane-admin` und `jane-s3only` entfernen, ohne die daraus resultierenden Berechtigungen zu ändern. Dadurch wird die Politik dynamisch. Es definiert Berechtigungen für alle zukünftigen Rollen, die mit `jane`. Wenn die Richtlinie den Zugriff auf eine statische Anzahl von Rollen zulassen soll, entfernen Sie den letzten ARN und listen Sie nur die ARNs auf, die definiert werden sollen.

- IAM-JSON-Richtlinienelemente: Resource

AWS verwaltete Richtlinien mit diesem Vorschlag

AWS Mit verwalteten Richtlinien können Sie beginnen, AWS indem Sie Berechtigungen auf der Grundlage allgemeiner AWS Anwendungsfälle zuweisen.

Redundante Ressourcen wirken sich nicht auf die Berechtigungen aus, die von der Richtlinie gewährt werden. Wenn Sie eine AWS verwaltete Richtlinie als Referenz für die Erstellung Ihrer vom Kunden verwalteten Richtlinie verwenden, AWS empfiehlt Ihnen, überflüssige Ressourcen aus Ihrer Richtlinie zu entfernen.

Vorschlag – Redundante Aussage

Das AWS Management Console Ergebnis dieser Prüfung enthält die folgende Meldung:

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Abruf eines Vorschlags

Das Element `Statement` ist das Hauptelement einer Richtlinie. Dieses Element ist erforderlich. Das `Statement`-Element kann eine einzelne Anweisung oder ein Array von einzelnen Anweisungen enthalten.

Wenn Sie dieselbe Anweisung mehrmals in eine lange Richtlinie einfügen, sind die Anweisungen redundant. Sie können eine der Anweisungen entfernen, ohne sich auf die von der Richtlinie erteilten Berechtigungen auswirken zu müssen. Wenn jemand eine Richtlinie bearbeitet, kann er eine der Anweisungen ändern, ohne das Duplikat zu aktualisieren. Dies kann zu mehr Berechtigungen führen als beabsichtigt.

- IAM-JSON-Richtlinienelemente: Statement

Vorschlag – Platzhalter im Servicenamen

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Abruf eines Vorschlags

Wenn Sie den Namen eines AWS Dienstes in eine Richtlinie aufnehmen, AWS empfiehlt es sich, keine Platzhalter (*,?) zu verwenden. Dadurch können Berechtigungen für zukünftige Services hinzugefügt werden, die Sie nicht beabsichtigen. Beispielsweise gibt es mehr als ein Dutzend AWS Dienste, deren Namen das Wort `*code*` enthalten.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- IAM-JSON-Richtlinienelemente: Resource

Vorschlag – Zulassen mit nicht unterstütztem Tag-Bedingungsschlüssel für Service

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
```

```
are not allowed by this statement. We recommend that you move these actions to a
different statement without this condition key.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
 affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Abruf eines Vorschlags

Die Verwendung von nicht unterstützten Tag-Bedingungsschlüsseln im `Condition` Element einer Richtlinie mit `"Effect": "Allow"` hat keinen Einfluss auf die durch die Richtlinie gewährten Berechtigungen, da die Bedingung für diese Dienstaktion ignoriert wird. AWS empfiehlt, die Aktionen für Dienste zu entfernen, die den Bedingungsschlüssel nicht unterstützen, und eine weitere Anweisung zu erstellen, um den Zugriff auf bestimmte Ressourcen in diesem Dienst zu ermöglichen.

Wenn Sie den `aws:ResourceTag`-Bedingungsschlüssel verwenden und dieser nicht von einer Service-Aktion unterstützt wird, wird der Schlüssel nicht in den Anforderungskontext aufgenommen. In diesem Fall gibt die Bedingung in der `Allow`-Anweisung immer `false` zurück und die Aktion ist nie erlaubt. Dies geschieht auch dann, wenn die Ressource korrekt markiert ist.

Wenn ein Service den Bedingungsschlüssel `aws:ResourceTag` unterstützt, können Sie den Zugriff auf die Ressourcen dieses Services mit Hilfe von Tags steuern. Dies wird als attributbasierte Zugriffskontrolle (ABAC) bezeichnet. Bei Services, die diese Schlüssel nicht unterstützen, müssen Sie den Zugriff auf Ressourcen mithilfe der ressourcenbasierten Zugriffskontrolle (RBAC) steuern.

> **Note**
>
> Einige Services ermöglichen die Unterstützung des Bedingungsschlüssel `aws:ResourceTag` für eine Teilmenge ihrer Ressourcen und Aktionen. IAM Access Analyzer gibt Ergebnisse für die Service-Aktionen zurück, die nicht unterstützt werden. Amazon S3 unterstützt beispielsweise `aws:ResourceTag` für eine Teilmenge der Ressourcen. Alle in Amazon S3 verfügbaren Ressourcentypen, die den `aws:ResourceTag`-Bedingungsschlüssel unterstützen, finden Sie unter Resource types defined by Amazon S3 in der Service Authorization Reference.

Nehmen Sie beispielsweise an, dass Sie den Zugriff auf bestimmte Ressourcen, die mit dem Schlüssel-Wert-Paar `team=BumbleBee` gekennzeichnet sind, verweigern möchten. Gehen Sie außerdem davon aus, dass Sie damit Ressourcen taggen AWS Lambda können, der `aws:ResourceTag` Bedingungsschlüssel jedoch nicht unterstützt wird. Verwenden Sie den `aws:ResourceTag` Bedingungsschlüssel, um Anzeigeaktionen für AWS App Mesh und AWS Backup falls dieses Tag vorhanden ist, zuzulassen. Verwenden Sie für Lambda eine Ressourcennamenskonvention, die den Teamnamen als Präfix enthält. Fügen Sie dann eine separate Anweisung hinzu, die das Anzeigen von Ressourcen mit dieser Namenskonvention ermöglicht.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

> **Warning**
>
> Verwenden Sie nicht die `Not` -Version des Bedingungsoperators mit `"Effect": "Allow"` als Abhilfe für diese Feststellung. Diese Bedingungsoperatoren bieten eine negierte Übereinstimmung. Dies bedeutet, dass nach der Auswertung der Bedingung

das Ergebnis negiert wird. Im vorigen Beispiel wird die Aktion `lambda:GetFunction` in der `AllowViewSupported`-Anweisung mit dem Operator `StringNotEquals` immer zugelassen, unabhängig davon, ob die Ressource mit einem Tag versehen ist. Verwenden Sie nicht die IfExistsVersion... des Bedingungsoperators, um dieses Problem zu umgehen. Das bedeutet: "Erlaube die Aktion, wenn der Schlüssel im Anforderungskontext vorhanden ist und die Werte übereinstimmen. Andernfalls erlauben Sie die Aktion". Im vorangegangenen Beispiel wurde die Aktion `lambda:GetFunction` mit dem Operator `AllowViewSupported` in die Anweisung `StringEqualsIfExists` aufgenommen, um die Aktion zu ermöglichen. Bei dieser Aktion ist der Schlüssel nicht im Kontext vorhanden, und jeder Versuch, diesen Ressourcentyp anzuzeigen, ist zulässig, unabhängig davon, ob die Ressource mit einem Tag versehen ist.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- Condition-Element
- Übersicht über JSON-Richtlinien

Vorschlag — Zulassen, wenn NotAction der Tag-Bedingungsschlüssel für den Service nicht unterstützt wird

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Abruf eines Vorschlags

Die Verwendung nicht unterstützter Tag-Bedingungsschlüssel im `Condition`-Element einer Richtlinie mit dem Element `NotAction` und `"Effect": "Allow"` hat keine Auswirkungen auf die von der Richtlinie gewährten Berechtigungen. Die Bedingung wird für Serviceaktionen ignoriert, die den Bedingungsschlüssel nicht unterstützen. AWS empfiehlt, die Logik neu zu schreiben, um eine Liste von Aktionen zuzulassen.

Wenn Sie den Bedingungsschlüssel `aws:ResourceTag` mit `NotAction` verwenden, werden alle neuen oder bestehenden Service-Aktionen, die den Schlüssel nicht unterstützen, nicht zugelassen. AWS empfiehlt, dass Sie die Aktionen, die Sie zulassen möchten, ausdrücklich auflisten. IAM Access Analyzer gibt eine separate Suche für aufgelistete Aktionen zurück, die die `aws:ResourceTag`-Bedingungsschlüssel. Weitere Informationen finden Sie unter Vorschlag – Zulassen mit nicht unterstütztem Tag-Bedingungsschlüssel für Service.

Wenn ein Service den Bedingungsschlüssel `aws:ResourceTag` unterstützt, können Sie den Zugriff auf die Ressourcen dieses Services mit Hilfe von Tags steuern. Dies wird als attributbasierte Zugriffskontrolle (ABAC) bezeichnet. Bei Services, die diese Schlüssel nicht unterstützen, müssen Sie den Zugriff auf Ressourcen mithilfe der ressourcenbasierten Zugriffskontrolle (RBAC) steuern.

Verwandte Begriffe

- Globale Bedingungsschlüssel
- ABAC mit RBAC vergleichen
- IAM-JSON-Richtlinienelemente: Bedingungsoperatoren
- Condition-Element
- Übersicht über JSON-Richtlinien

Vorschlag - Empfohlener Bedingungsschlüssel für Service-Prinzipal

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Abruf eines Vorschlags

Sie können AWS-Services im `Principal` Element einer ressourcenbasierten Richtlinie einen Dienstprinzipal angeben, bei dem es sich um einen Bezeichner für den Dienst handelt. Sie sollten die Bedingungsschlüssel `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` oder `aws:SourceOrgPaths` verwenden, wenn Sie Zugriff auf Service-Prinzipale gewähren, anstelle anderer Bedingungsschlüssel wie `aws:Referer`. Dies hilft Ihnen, ein Sicherheitsproblem zu verhindern, das Problem des verwirrten Stellvertreters genannt wird.

Verwandte Begriffe

- AWS-Service Prinzipale
- AWS globale Bedingungsschlüssel: aws: SourceAccount
- AWS globale Bedingungsschlüssel: aws: SourceArn
- AWS globale Bedingungsschlüssel: aws: SourceOrgId
- AWS globale Bedingungsschlüssel: aws: SourceOrgPaths
- Das Confused-Deputy-Problem

Vorschlag - Irrelevanter Bedingungsschlüssel in der Richtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Abruf eines Vorschlags

Einige Bedingungsschlüssel sind für ressourcenbasierte Richtlinien nicht relevant. Zum Beispiel, der `s3:ResourceAccount`-Bedingungsschlüssel ist nicht relevant für die ressourcenbasierte Richtlinie, die an einen Amazon-S3-Bucket oder Amazon-S3-Zugriffspunkt-Ressourcentyp angehängt ist.

Sie sollten Ihren Bedingungsschlüssel in einer identitätsbasierten Richtlinie verwenden, um den Zugriff auf die Ressource zu kontrollieren.

Verwandte Begriffe

- Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien

Vorschlag – Redundanter Prinzipal in Rollenvertrauensrichtlinie

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Abruf eines Vorschlags

Wenn Sie sowohl einen Prinzipal mit angenommener Rolle als auch seine übergeordnete Rolle im `Principal`–Element einer Richtlinie angeben, erlaubt oder verweigert sie keine anderen Berechtigungen. Zum Beispiel ist es überflüssig, das `Principal`-Element im folgenden Format anzugeben:

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Wir empfehlen, den Prinzip mit angenommener Rolle zu entfernen.

Verwandte Begriffe

- Rollensitzungsgsprinzipale

Vorschlag – Bestätigung des Zielgruppenanspruch-Typs

In der AWS Management Console beinhaltet das Ergebnis dieser Prüfung die folgende Meldung:

```
Confirm audience claim type: The 'aud' (audience) claim key identifies the recipients
 that the JSON web token is intended for. Audience claims can be multivalued or single-
valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If
 the claim is single-valued, do not use a qualifier.
```

Bei programmatischen Aufrufen der AWS CLI AWS OR-API enthält das Ergebnis dieser Prüfung die folgende Meldung:

```
"findingDetails": "The 'aud' (audience) claim key identifies the recipients that the
 JSON web token is intended for. Audience claims can be multivalued or single-valued.
 If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the claim
 is single-valued, do not use a qualifier."
```

Abruf eines Vorschlags

Der aud-Anspruchsschlüssel (Zielgruppe) ist ein eindeutiger Bezeichner für Ihre App, der ausgestellt wird, wenn Sie Ihre App beim IdP registrieren. Er identifiziert die Empfänger, für die das JSON-Web-Token bestimmt ist. Zielgruppenansprüche können mehrwertig oder einwertig sein. Wenn der Anspruch mehrwertig ist, verwenden Sie einen `ForAllValues`- oder `ForAnyValue`-Bedingungssatz-Operator. Wenn der Anspruch einwertig ist, verwenden Sie keinen Bedingungssatz-Operator.

Verwandte Begriffe

- Erstellen einer Rolle für Web-Identität oder OpenID-Connect-Verbund (Konsole)
- Mehrwertige Kontextschlüssel
- Einzelwertige vs. mehrwertige Bedingungsschlüssel

IAM Access Analyzer – Benutzerdefinierte Richtlinienüberprüfungen

Sie können Ihre Richtlinien mithilfe von benutzerdefinierten Richtlinienüberprüfungen von AWS Identity and Access Management Access Analyzer mit Ihren vorhandenen Sicherheitsstandards abgleichen. Sie können die folgenden Arten von benutzerdefinierten Richtlinienprüfungen ausführen:

- Vergleich mit einer Referenzrichtlinie: Wenn Sie eine Richtlinie bearbeiten, können Sie überprüfen, ob die aktualisierte Richtlinie im Gegensatz zu einer Referenzrichtlinie Zugang gewährt. Zum Vergleich könnten Sie beispielsweise eine vorhandene Version der Richtlinie heranziehen. Sie können diese Prüfung ausführen, wenn Sie eine Richtlinie mit der AWS Command Line Interface (AWS CLI), der IAM Access Analyzer API (API) oder dem JSON-Richtlinieneditor in der IAM-Konsole bearbeiten.
- Mit einer Liste von IAM-Aktionen oder -Ressourcen vergleichen: Sie können überprüfen, ob bestimmte IAM-Aktionen oder -Ressourcen gemäß Ihrer Richtlinie nicht zulässig sind. Wenn nur Aktionen angegeben sind, prüft IAM Access Analyzer, ob die Aktionen auf alle Ressourcen in der Richtlinie zugegriffen haben. Wenn nur Ressourcen angegeben sind, überprüft IAM Access Analyzer, welche Aktionen Zugriff auf die angegebenen Ressourcen haben. Wenn sowohl Aktionen als auch Ressourcen angegeben sind, überprüft IAM Access Analyzer, welche der angegebenen Aktionen Zugriff auf die angegebenen Ressourcen haben. Sie können diese Prüfung ausführen, wenn Sie eine Richtlinie mithilfe der AWS CLI oder der API erstellen oder bearbeiten.
- Auf öffentlichen Zugriff prüfen: Sie können überprüfen, ob eine Ressourcenrichtlinie öffentlichen Zugriff auf einen bestimmten Ressourcentyp gewähren kann. Sie können diese Prüfung ausführen, wenn Sie eine Richtlinie mithilfe der AWS CLI oder der API erstellen oder bearbeiten. Diese Art der Überprüfung benutzerdefinierter Richtlinien unterscheidet sich von der Zugriffsvorschau, da für die Prüfung kein Konto oder ein externer Zugriffsanalysekontext erforderlich ist. Mithilfe von Access-Vorschauen können Sie eine Vorschau der Ergebnisse von IAM Access Analyzer anzeigen, bevor Sie Ressourcenberechtigungen bereitstellen, während die benutzerdefinierte Prüfung bestimmt, ob der öffentliche Zugriff möglicherweise durch eine Richtlinie gewährt wird.

Mit jeder Überprüfung der benutzerdefinierten Richtlinien ist eine Gebühr verbunden. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

So funktionieren benutzerdefinierte Richtlinienüberprüfungen

Sie können benutzerdefinierte Richtlinienüberprüfungen für identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien ausführen. Benutzerdefinierte Richtlinienüberprüfungen basieren nicht auf Methoden wie dem Musterabgleich oder der Untersuchung von Zugriffsprotokollen, um festzustellen, ob ein neuer oder ein bestimmter Zugriff gemäß einer Richtlinie zulässig ist. Ähnlich wie bei externen Zugriffsergebnissen basieren benutzerdefinierte Richtlinienüberprüfungen auf Zelkova. Zelkova übersetzt IAM-Richtlinien in äquivalente logische Anweisungen und wendet eine Reihe von allgemeinen und spezialisierten logischen Solvern (Erfüllbarkeits-Modulo-Theorien) auf das Problem an. Um nach neuen oder bestimmten Zugriffen zu suchen, wendet IAM Access Analyzer Zelkova wiederholt auf eine Richtlinie an. Abfragen werden immer spezifischer, um Verhaltensklassen

zu beschreiben, die die Richtlinie basierend ihrem Inhalt zulässt. Weitere Informationen zu den Satisfiability Modulo-Theorien finden Sie unter Satisfiability Modulo-Theorien.

In seltenen Fällen kann IAM Access Analyzer nicht vollständig feststellen, ob eine Richtlinienanweisung neuen oder bestimmten Zugriff gewährt. In diesen Fällen wird tendenziell ein falsch positives Ergebnis gemeldet, indem die benutzerdefinierte Richtlinienüberprüfung scheitert. IAM Access Analyzer soll eine umfassende Richtlinienbewertung ermöglichen und die Menge an falsch negativen Ergebnissen minimieren. Mit diesem Ansatz bietet IAM Access Analyzer ein hohes Maß an Sicherheit; eine bestandene Prüfung bedeutet, dass der Zugriff durch die Richtlinie nicht gewährt wurde. Sie können fehlgeschlagene Prüfungen manuell kontrollieren, indem Sie die Richtlinienanweisung überprüfen, die in der Antwort von IAM Access Analyzer angegeben ist.

Beispiele für Referenzrichtlinien zum Prüfen auf neue Zugriffe

Beispiele für Referenzrichtlinien und Informationen zum Einrichten und Ausführen einer benutzerdefinierten Richtlinienprüfung für neuen Zugriff finden Sie im IAM Access Analyzer-Beispiel-Repository für benutzerdefinierte Richtlinienprüfungen. GitHub

Hinweise zur Verwendung dieser Beispiele

Führen Sie die folgenden Schritte aus, bevor Sie diese Beispiel-Referenzrichtlinien verwenden:

- Überprüfen Sie die Referenzrichtlinien sorgfältig und passen Sie sie an Ihre individuellen Anforderungen an.
- Testen Sie die Referenzrichtlinien in Ihrer Umgebung gründlich mit den von Ihnen verwendeten AWS-Services .

  Die Referenzrichtlinien veranschaulichen die Implementierung und Verwendung von benutzerdefinierten Richtlinienüberprüfungen. Sie sind nicht als offizielle Empfehlungen von AWS oder bewährte Methoden zu interpretieren, die genau wie gezeigt umgesetzt werden müssen. Es liegt in Ihrer Verantwortung, alle Referenzrichtlinien sorgfältig zu testen, ob sie die Sicherheitsanforderungen Ihrer Umgebung zu erfüllen.
- Benutzerdefinierte Richtlinienüberprüfungen sind in ihrer Analyse umgebungsunabhängig. Bei ihrer Analyse werden nur Informationen berücksichtigt, die in den Eingaberichtlinien enthalten sind. Mit benutzerdefinierten Richtlinienprüfungen kann beispielsweise nicht überprüft werden, ob ein Konto Mitglied einer bestimmten AWS Organisation ist. Daher können die benutzerdefinierten Richtlinienüberprüfungen neue Zugriffe nicht anhand der

Bedingungsschlüsselwerte für die Bedingungsschlüssel `aws:PrincipalOrgId` und `aws:PrincipalAccount` vergleichen.

Überprüfung fehlgeschlagener benutzerdefinierter Richtlinien

Wenn eine benutzerdefinierte Richtlinienüberprüfung fehlschlägt, enthält die Antwort von IAM Access Analyzer die Anweisungs-ID (`Sid`) der Richtlinienanweisung, die zum Fehlschlagen der Prüfung geführt hat. Obwohl es sich bei der Anweisungs-ID um ein optionales Richtlinienelement handelt, empfehlen wir, dass Sie für jede Richtlinienanweisung eine Anweisungs-ID hinzufügen. Die benutzerdefinierte Richtlinienüberprüfung gibt auch einen Anweisungsindex zurück, anhand dessen der Grund für die fehlgeschlagene Prüfung ermittelt werden kann. Der Anweisungsindex folgt einer auf Null basierenden Nummerierung, wobei auf die erste Anweisung die 0 hat. Wenn mehrere Anweisungen dazu führen, dass eine Prüfung fehlschlägt, gibt die Prüfung jeweils nur eine Anweisungs-ID zurück. Wir empfehlen Ihnen, die in der Begründung hervorgehobene Anweisung zu korrigieren und die Prüfung erneut durchzuführen, bis sie erfolgreich ist.

Überprüfen von Richtlinien mit benutzerdefinierten Richtlinienüberprüfungen (Konsole)

Als optionalen Schritt können Sie eine benutzerdefinierte Richtlinienüberprüfung ausführen, wenn Sie eine Richtlinie im JSON-Richtlinieneditor in der IAM-Konsole bearbeiten. Sie können überprüfen, ob die aktualisierte Richtlinie im Gegensatz zur vorhandenen Version nun Zugriff gewährt.

So suchen Sie bei der Bearbeitung von IAM-JSON-Richtlinien nach neuen Zugriffen

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).
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie bearbeiten möchten. Sie können über das Suchfeld die Liste der Gruppen filtern.
4. Wählen Sie die Registerkarte Berechtigungen und anschließend Richtlinie bearbeiten aus.
5. Wählen Sie die Option JSON und aktualisieren Sie Ihre Richtlinie.
6. Wählen Sie im Bereich der Richtlinienüberprüfung unterhalb der Richtlinie die Registerkarte Nach neuem Zugriff prüfen und dann Richtlinie überprüfen. Wenn die geänderten Berechtigungen bisher nicht gewährten Zugriff gewähren, wird die Anweisung im Bereich zur Richtlinienvalidierung hervorgehoben.

7. Wenn Sie nicht beabsichtigen, neuen Zugriff zu gewähren, aktualisieren Sie die Richtlinienanweisung und wählen Sie Richtlinie überprüfen, bis kein neuer Zugriff erkannt wird.

> **Note**
>
> Für jede Prüfung auf bisher nicht gewährten Zugriff wird eine Gebühr erhoben. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für IAM Access Analyzer.

8. Wählen Sie Weiter aus.
9. Überprüfen Sie auf der Seite Überprüfen und Speichern den Bereich In dieser Richtlinie definierte Berechtigungen und wählen Sie dann Änderungen speichern aus.

Überprüfen von Richtlinien mit benutzerdefinierten Richtlinienprüfungen (AWS CLI oder API)

Sie können benutzerdefinierte IAM Access Analyzer-Richtlinienprüfungen über die AWS CLI oder die IAM Access Analyzer-API ausführen.

So führen Sie benutzerdefinierte Richtlinienüberprüfungen mit IAM Access Analyzer durch (AWS CLI)

- Führen Sie den folgenden Befehl aus, um zu überprüfen, ob für eine aktualisierte Richtlinie im Vergleich zur vorhandenen Richtlinie neuer Zugriff gewährt wird: check-no-new-access
- Führen Sie den folgenden Befehl aus, um zu überprüfen, ob der bestimmte Zugriff durch eine Richtlinie nicht zulässig ist: check-access-not-granted
- Führen Sie den folgenden Befehl aus, um zu überprüfen, ob eine Ressourcenrichtlinie öffentlichen Zugriff auf einen bestimmten Ressourcentyp gewähren kann: check-no-public-access

So führen Sie benutzerdefinierte Richtlinienüberprüfungen mit IAM Access Analyzer durch (API)

- Nutzen Sie die API-Operation CheckNoNewAccess, um zu überprüfen, ob für eine aktualisierte Richtlinie im Vergleich zur vorhandenen Richtlinie neuer Zugriff gewährt wird.
- Verwenden Sie die API-Operation CheckAccessNotGranted, um zu überprüfen, ob der angegebene Zugriff durch eine Richtlinie nicht zulässig ist.
- Verwenden Sie den CheckNoPublicAccessAPI-Vorgang, um zu überprüfen, ob eine Ressourcenrichtlinie öffentlichen Zugriff auf einen bestimmten Ressourcentyp gewähren kann.

Generieren von IAM Access Analyzer-Richtlinien

Als Administrator oder Entwickler können Sie IAM-Entitäten (Benutzern oder Rollen) Berechtigungen gewähren, die über das hinausgehen, was sie benötigen. IAM bietet verschiedene Optionen, mit denen Sie die von Ihnen erteilten Berechtigungen verfeinern können. Eine Option besteht darin, eine IAM-Richtlinie zu generieren, die auf der Zugriffsaktivität für eine Entität basiert. IAM Access Analyzer überprüft Ihre AWS CloudTrail Protokolle und generiert eine Richtlinienvorlage, die die Berechtigungen enthält, die die Entität in Ihrem angegebenen Zeitraum verwendet hat. Sie können die Vorlage verwenden, um eine Richtlinie mit definierten Berechtigungen zu erstellen, die nur die Berechtigungen gewährt, die zur Unterstützung Ihres spezifischen Anwendungsfalls erforderlich sind.

Themen

So funktioniert die Richtlinienerzeugung

IAM Access Analyzer analysiert Ihre CloudTrail Ereignisse, um Aktionen und Dienste zu identifizieren, die von einer IAM-Entität (Benutzer oder Rolle) verwendet wurden. Es generiert dann eine IAM-Richtlinie, die auf dieser Aktivität basiert. Sie können die Berechtigungen einer Entität verfeinern, wenn Sie eine umfassende, an die Entität angefügte Berechtigungsrichtlinie durch die generierte Richtlinie ersetzen. Nachfolgend finden Sie einen allgemeinen Überblick über den Prozess der Richtliniengenerierung.

- Einrichtung für die Generierung von Richtlinienvorlagen — Sie geben einen Zeitraum von bis zu 90 Tagen an, in dem IAM Access Analyzer Ihre historischen Ereignisse analysieren soll. AWS CloudTrail Sie müssen eine vorhandene Servicerolle angeben oder eine neue erstellen. Die Service-Rolle gewährt IAM Access Analyzer Zugriff auf Ihre CloudTrail Trail- und Service-

Informationen, auf die zuletzt zugegriffen wurde, um die Dienste und Aktionen zu identifizieren, die verwendet wurden. Sie müssen den CloudTrail Trail angeben, der Ereignisse für das Konto protokolliert, bevor Sie eine Richtlinie generieren können. Weitere Informationen zu IAM Access CloudTrail Analyzer-Datenkontingenten finden Sie unter IAM Access Analyzer-Kontingente.

- Richtlinie generieren — IAM Access Analyzer generiert eine Richtlinie, die auf der Zugriffsaktivität Ihrer Ereignisse basiert. CloudTrail
- Richtlinie überprüfen und anpassen – Nachdem die Richtlinie generiert wurde, können Sie die Services und Aktionen überprüfen, die von der Entität während des angegebenen Datumsbereichs verwendet wurden. Sie können die Richtlinie weiter anpassen, indem Sie Berechtigungen hinzufügen oder entfernen, Ressourcen angeben und der Richtlinienvorlage Bedingungen hinzufügen.
- Richtlinie erstellen und anhängen – Sie haben die Möglichkeit, die generierte Richtlinie zu speichern, indem Sie eine verwaltete Richtlinie erstellen. Sie können die von Ihnen erstellte Richtlinie an den Benutzer oder die Rolle anhängen, deren Aktivität zum Generieren der Richtlinie verwendet wurde.

Informationen auf Service- und Aktionsebene

Wenn IAM Access Analyzer eine IAM-Richtlinie generiert, werden Informationen zurückgegeben, die Ihnen helfen, die Richtlinie weiter anzupassen. Bei der Generierung einer Richtlinie können zwei Kategorien von Informationen zurückgegeben werden:

- Richtlinie mit Informationen auf Aktionsebene — Für einige AWS Services, wie Amazon EC2, kann IAM Access Analyzer die in Ihren CloudTrail Ereignissen gefundenen Aktionen identifizieren und die Aktionen auflisten, die in der generierten Richtlinie verwendet werden. Eine Liste der unterstützten Services finden Sie unter IAM Access Analyzer Services zur Richtliniengenerierung. Bei einigen Services fordert IAM Access Analyzer Sie auf, der generierten Richtlinie Aktionen für die Services hinzuzufügen.
- Richtlinie mit Informationen auf Serviceebene – IAM Access Analyzer verwendet Informationen, auf die zuletzt zugegriffen wurde, um eine Richtlinienvorlage mit allen kürzlich verwendeten Services zu erstellen. Wenn Sie die verwenden AWS Management Console, fordern wir Sie auf, die Services zu überprüfen und Aktionen hinzuzufügen, um die Richtlinie zu vervollständigen.

Eine Liste der Aktionen in den einzelnen Diensten finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste in der Serviceautorisierungsreferenz.

Wissenswerte Informationen zur Erstellung von Richtlinien

Bevor Sie eine Richtlinie erstellen, lesen Sie die folgenden wichtigen Details.

- Einen CloudTrail Trail aktivieren — Sie müssen einen CloudTrail Trail für Ihr Konto aktiviert haben, um eine Richtlinie auf der Grundlage der Zugriffsaktivität erstellen zu können. Wenn Sie einen CloudTrail Trail erstellen, CloudTrail sendet Ereignisse, die sich auf Ihren Trail beziehen, an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Informationen zum Erstellen eines CloudTrail Trails finden Sie unter Erstellen eines Trails für Ihr AWS Konto im AWS CloudTrail Benutzerhandbuch.
- Datenereignisse nicht verfügbar – IAM Access Analyzer identifiziert keine Aktivitäten auf Aktionsebene für Datenereignisse, wie z. B. Amazon S3 Datenereignisse, in generierten Richtlinien.
- PassRole— Die `iam:PassRole` Aktion wird von den generierten Richtlinien nicht nachverfolgt CloudTrail und ist auch nicht in diesen enthalten.
- Reduzieren der Zeit zur Erstellung von Richtlinien – Um eine Richtlinie schneller zu generieren, reduzieren Sie den Datumsbereich, den Sie während der Einrichtung für die Richtliniengenerierung angeben.
- CloudTrail Für Auditzwecke verwenden — Verwenden Sie die Richtliniengenerierung nicht für Prüfungszwecke, CloudTrail sondern verwenden Sie. Weitere Informationen zur Verwendung CloudTrail finden Sie unter Protokollieren von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail.
- Abgelehnte Aktionen — Bei der Richtliniengenerierung werden alle CloudTrail Ereignisse überprüft, auch abgelehnte Aktionen.
- IAM-Konsole für eine Richtlinie – Sie können jeweils eine generierte Richtlinie in der IAM-Konsole haben.
- IAM-Konsole für generierte Richtlinienverfügbarkeit – Sie können eine generierte Richtlinie in der IAM-Konsole bis zu 7 Tage lang überprüfen, nachdem sie generiert wurde. Nach 7 Tagen müssen Sie eine neue Richtlinie erstellen.
- Kontingente zur Generierung von Richtlinien – Weitere Informationen zu IAM Access Analyzer-Richtliniengenerierungs-Kontingente finden Sie unter IAM Access Analyzer-Kontingente.
- Es gelten die Standardtarife von Amazon S3 — Wenn Sie die Funktion zur Richtliniengenerierung verwenden, überprüft IAM Access Analyzer die CloudTrail Protokolle in Ihrem S3-Bucket. Für den Zugriff auf Ihre CloudTrail Protokolle zur Richtliniengenerierung fallen keine zusätzlichen

Speichergebühren an. AWS berechnet die Amazon S3 S3-Standardtarife für Anfragen und die Datenübertragung von CloudTrail Protokollen, die in Ihrem S3-Bucket gespeichert sind.

- AWS Control Tower Unterstützung — Bei der Richtliniengenerierung wird die AWS Control Tower Generierung von Richtlinien nicht unterstützt.

Erforderliche Berechtigungen zum Generieren einer Richtlinie

Die Berechtigungen, die Sie zum ersten Mal benötigen, um eine Richtlinie zu generieren, unterscheiden sich von denen, die Sie benötigen, um eine Richtlinie für nachfolgende Verwendungen zu generieren.

Erstmalige Einrichtung

Wenn Sie zum ersten Mal eine Richtlinie generieren, müssen Sie eine geeignete vorhandene Servicerolle in Ihrem Konto auswählen oder eine neue Servicerolle erstellen. Die Servicerolle gewährt IAM Access Analyzer Zugriff auf die Informationen CloudTrail und die Informationen, auf die der Dienst zuletzt zugegriffen hat, in Ihrem Konto. Nur Administratoren sollten über die erforderlichen Berechtigungen verfügen, um Rollen zu erstellen und zu konfigurieren. Daher empfehlen wir, dass ein Administrator die Servicerolle während der Ersteinrichtung erstellt. Weitere Informationen zu den Berechtigungen, die zum Erstellen von Servicerollen erforderlich sind, finden Sie unter Eine Rolle erstellen, um Berechtigungen für einen AWS Dienst zu delegieren.

Erforderliche Berechtigungen für Servicerollen

Wenn Sie eine Servicerolle erstellen, konfigurieren Sie zwei Richtlinien für die Rolle. Sie fügen eine IAM-Berechtigungsrichtlinie an die Rolle an, die angibt, was mit der Rolle getan werden kann. Außerdem fügen Sie der Rolle eine Rollenvertrauensrichtlinie hinzu, die den Prinzipal angibt, der die Rolle verwenden kann.

Die erste Beispielrichtlinie zeigt die Berechtigungsrichtlinie für die Servicerolle, die zum Generieren einer Richtlinie erforderlich ist. Die zweite Beispielrichtlinie zeigt die Rollenvertrauensrichtlinie, die für die Servicerolle erforderlich ist. Sie können diese Richtlinien verwenden, um eine Servicerolle zu erstellen, wenn Sie die AWS API verwenden, oder AWS CLI um eine Richtlinie zu generieren. Wenn Sie die IAM-Konsole verwenden, um eine Servicerolle im Rahmen des Richtliniengenerierungsprozesses zu erstellen, generieren wir diese Richtlinien für Sie.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

Die folgende Beispielrichtlinie zeigt die Rollenvertrauensrichtlinie mit den Berechtigungen, die es IAM Access Analyzer ermöglichen, die Rolle zu übernehmen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Nachfolgende Verwendungen

Um Richtlinien in der zu generieren AWS Management Console, muss ein IAM-Benutzer über eine Berechtigungsrichtlinie verfügen, die es ihm ermöglicht, die für die Richtliniengenerierung verwendete Servicerolle an IAM Access Analyzer zu übergeben. `iam:PassRole`wird normalerweise von beigefügt, `iam:GetRole` damit der Benutzer die Details der zu übergebenden Rolle abrufen kann. In diesem Beispiel kann der Benutzer nur Rollen übergeben, die im angegebenen Konto mit Namen vorhanden sind, die mit `AccessAnalyzerMonitorServiceRole*` beginnen. Weitere Informationen zur Übergabe von IAM-Rollen an AWS Dienste finden Sie unter Einem Benutzer Berechtigungen zur Übergabe einer Rolle an einen AWS Dienst erteilen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

Sie benötigen außerdem die folgenden IAM Access Analyzer-Berechtigungen, um Richtlinien in der AWS Management Console, AWS API oder AWS CLI wie in der folgenden Richtlinienerklärung beschrieben generieren zu können.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
```

```
}
```

Sowohl für Erst- als auch für spätere Anwendungen

Wenn Sie die AWS Management Console zum Generieren einer Richtlinie verwenden, benötigen Sie die `cloudtrail:ListTrails` Berechtigung, die CloudTrail Trails in Ihrem Konto aufzulisten, wie in der folgenden Richtlinienerklärung beschrieben.

```
{
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "CloudTrail:ListTrails"
  ],
  "Resource": "*"
}
```

Generieren Sie eine Richtlinie auf der Grundlage von CloudTrail Aktivitäten (Konsole)

Sie können eine Richtlinie für einen IAM-Benutzer oder eine Rolle generieren.

Schritt 1: Generieren Sie eine Richtlinie auf der Grundlage von CloudTrail Aktivitäten

Im folgenden Verfahren wird erläutert, wie Sie mithilfe des AWS Management Console eine Richtlinie für eine Rolle generieren.

Generieren einer Richtlinie für eine IAM-Rolle

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 Roles (Rollen).

   > **Note**
   >
   > Die Schritte zum Generieren einer Richtlinie basierend auf Aktivitäten für einen IAM-Benutzer sind nahezu identisch. Wählen Sie dazu Benutzer anstelle von Rollen.

3. Wählen Sie in der Rollenliste in Ihrem Konto den Namen der Rolle aus, deren Aktivität Sie zum Generieren einer Richtlinie verwenden möchten.

4. Wählen Sie auf der Registerkarte Berechtigungen im Abschnitt Richtlinie basierend auf CloudTrail Ereignissen generieren die Option Richtlinie generieren aus.
5. Geben Sie auf der Seite Richtlinie generieren den Zeitraum an, in dem IAM Access Analyzer Ihre CloudTrail Ereignisse auf Aktionen hin analysieren soll, die mit der Rolle ergriffen wurden. Sie können einen Bereich von bis zu 90 Tagen wählen. Wir empfehlen Ihnen, den kürzest möglichen Zeitraum zu wählen, um die Zeit für die Generierung von Richtlinien zu verkürzen.
6. Wählen Sie im Bereich CloudTrail Zugriff eine geeignete bestehende Rolle aus, oder erstellen Sie eine neue Rolle, falls keine passende Rolle vorhanden ist. Die Rolle gibt IAM Access Analyzer die Erlaubnis, in Ihrem Namen auf Ihre CloudTrail Daten zuzugreifen, um die Zugriffsaktivitäten zu überprüfen und die Dienste und Aktionen zu identifizieren, die verwendet wurden. Verwenden Sie Erforderliche Berechtigungen zum Generieren einer Richtlinie, um mehr über die für diese Rolle erforderlichen Berechtigungen zu erfahren.
7. Geben Sie im Abschnitt Zu analysierende CloudTrail CloudTrail Spur den Pfad an, in dem Ereignisse für das Konto protokolliert werden.

   Wenn Sie einen CloudTrail Trail wählen, der Logs in einem anderen Konto speichert, wird ein Informationsfeld über den kontoübergreifenden Zugriff angezeigt. Kontenübergreifender Zugriff erfordert eine zusätzliche Einrichtung. Weitere Informationen finden Sie unter Choose a role for cross-account access weiter unten in diesem Thema.
8. Wählen Sie Richtlinie generieren.
9. Während die Richtliniengenerierung im Gange ist, kehren Sie zur Seite Rollen-Übersicht auf der Registerkarte Berechtigungen zurück. Warten Sie, bis der Status im Abschnitt Details zur Richtlinienanforderung Erfolg anzeigt, und wählen Sie dann Generierte Policy anzeigen. Sie können die generierte Richtlinie bis zu 7 Tage lang einsehen. Wenn Sie eine andere Richtlinie generieren, wird die vorhandene Richtlinie durch die neue ersetzt, die Sie generieren.

Schritt 2: Überprüfen der Berechtigungen und Hinzufügen von Aktionen für verwendete Services

Überprüfen Sie die Services und Aktionen, die IAM Access Analyzer als die verwendete Rolle identifiziert hat. Sie können Aktionen für alle Services hinzufügen, die für die generierte Richtlinienvorlage verwendet wurden.

1. Lesen Sie die folgenden Abschnitte:

- Überprüfen Sie auf der Seite Berechtigungen überprüfen die Liste der Aktionen, die in der generierten Richtlinie enthalten sind. Die Liste zeigt die Services und Aktionen an, die von IAM Access Analyzer identifiziert wurden und die von der Rolle im angegebenen Datumsbereich verwendet wurden.
- Im Abschnitt Verwendete Services werden zusätzliche Services angezeigt, die von IAM Access Analyzer identifiziert wurden und die von der Rolle im angegebenen Zeitraum verwendet wurden. Informationen darüber, welche Aktionen verwendet wurden, stehen für die in diesem Abschnitt aufgeführten Services möglicherweise nicht zur Verfügung. Verwenden Sie die Menüs für jedes aufgeführte Service, um die Aktionen, die Sie in die Richtlinie aufnehmen möchten, manuell auszuwählen.

2. Wenn Sie mit dem Hinzufügen von Aktionen fertig sind, wählen Sie Nächstes.

Schritt 3: Weiteres Anpassen der generierten Richtlinie

Sie können die Richtlinie weiter anpassen, indem Sie Berechtigungen hinzufügen oder entfernen oder Ressourcen angeben.

Anpassen der generierten Richtlinie

1. Aktualisieren Sie die Richtlinienvorlage. Die Richtlinienvorlage enthält ARN-Platzhalter für Ressourcen für Aktionen, die Berechtigungen auf Ressourcenebene unterstützen, wie in der folgenden Abbildung dargestellt. Berechtigungen auf Ressourcenebene bedeutet, dass Sie angeben können, für welche Ressourcen die Benutzer Aktionen ausführen dürfen. Wir empfehlen, dass Sie ARNs verwenden, um Ihre individuellen Ressourcen in der Richtlinie für Aktionen anzugeben, die Berechtigungen auf Ressourcenebene unterstützen. Sie können die Platzhalter-Ressourcen-ARNs durch gültige Ressourcen-ARNs für Ihren Anwendungsfall ersetzen.

   Wenn eine Aktion keine Berechtigungen auf Ressourcenebene unterstützt, müssen Sie einen Platzhalter (*) verwenden, um anzugeben, dass alle Ressourcen von der Aktion betroffen sein können. Informationen darüber, welche AWS Dienste Berechtigungen auf Ressourcenebene unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Eine Liste der Aktionen in jedem Service und um zu erfahren, welche Aktionen Berechtigungen auf Ressourcenebene unterstützen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS - Services.

2. (Optional) Fügen Sie JSON-Richtlinienanweisungen in der Vorlage hinzu, ändern oder entfernen Sie sie. Weitere Informationen zum Schreiben von JSON-Richtlinien finden Sie unter Erstellen von IAM-Richtlinien (Konsole).
3. Wenn Sie mit dem Anpassen der Richtlinienvorlage fertig sind, haben Sie die folgenden Optionen:

   - (Optional) Sie können das JSON in die Vorlage kopieren, um es separat außerhalb der Seite Generierte Richtlinie zu verwenden. Zum Beispiel, wenn Sie den JSON verwenden möchten, um eine Richtlinie in einem anderen Konto zu erstellen. Wenn die Richtlinie in Ihrer Vorlage das Limit von 6.144 Zeichen für JSON-Richtlinien überschreitet, ist die Richtlinie in mehrere Richtlinien unterteilt.
   - Wählen Sie Nächstes, um eine verwaltete Richtlinie im selben Konto zu überprüfen und zu erstellen.

Schritt 4: Überprüfen und Erstellen einer verwalteten Richtlinie

Wenn Sie über Berechtigungen zum Erstellen und Anhängen von IAM-Richtlinien verfügen, können Sie eine verwaltete Richtlinie aus der generierten Richtlinie erstellen. Sie können die Richtlinie dann an einen Benutzer oder eine Rolle in Ihrem Konto anhängen.

Überprüfen und Erstellen einer Richtlinie

1. Geben Sie auf der Seite Review and create managed policy (Überprüfen und Erstellen einer verwalteten Richtlinie) unter Name einen Namen und unter Description (Beschreibung) (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen.
2. (Optional) Im Abschnitt Zusammenfassung können Sie die Berechtigungen überprüfen, die in der Richtlinie enthalten sein werden.
3. (Optional) Fügen Sie der Richtlinie Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags mit IAM finden Sie unter Tagging von Amazon RDSIAM-Ressourcen.
4. Wenn Sie fertig sind, führen Sie einen der folgenden Schritte aus:

   - Sie können die neue Richtlinie direkt an die Rolle anhängen, die zum Generieren der Richtlinie verwendet wurde. *Aktivieren Sie dazu unten auf der Seite das Kontrollkästchen neben der* ***Richtlinie an den Namen anhängen.*** *YourRole* Wählen Sie dann Erstellen und Anfügen von Richtlinien.
   - Wählen Sie andernfalls Richtlinie erstellen. Die von Ihnen erstellte Richtlinie finden Sie in der Liste der Richtlinien im Navigationsbereich Richtlinien der IAM-Konsole.
5. Sie können die von Ihnen erstellte Richtlinie an eine Entität in Ihrem Konto anhängen. Nachdem Sie die Richtlinie angehängt haben, können Sie alle anderen übermäßig breiten Richtlinien entfernen, die möglicherweise an die Entität angehängt sind. Informationen zum Anhängen einer verwalteten Richtlinie finden Sie unter IAM-Identitätsberechtigungen hinzufügen (Konsole).

Generieren Sie eine Richtlinie mithilfe von AWS CloudTrail Daten in einem anderen Konto

Sie könnten CloudTrail Pfade erstellen, in denen Daten in zentralen Konten gespeichert werden, um die Verwaltungsaktivitäten zu optimieren. Sie können dies beispielsweise verwenden, um einen Pfad AWS Organizations zu erstellen, in dem alle Ereignisse für alle Personen AWS-Konten in dieser Organisation protokolliert werden. Der Trail gehört zu einem zentralen Konto. Wenn Sie eine Richtlinie für einen Benutzer oder eine Rolle in einem Konto generieren möchten, das sich von dem Konto unterscheidet, in dem Ihre CloudTrail Protokolldaten gespeichert sind, müssen Sie kontoübergreifenden Zugriff gewähren. Dazu benötigen Sie sowohl eine Rollen- als auch eine Bucket-Richtlinie, die IAM Access Analyzer-Berechtigungen für Ihre CloudTrail Protokolle gewähren. Weitere Informationen zum Erstellen von Organizations-Trails finden Sie unter Erstellen eines Trails für eine Organisation.

Gehen Sie in diesem Beispiel davon aus, dass Sie eine Richtlinie für einen Benutzer oder eine Rolle in Konto A generieren möchten. Der CloudTrail Trail in Konto A speichert CloudTrail Logs in einem Bucket in Konto B. Bevor Sie eine Richtlinie generieren können, müssen Sie die folgenden Aktualisierungen vornehmen:

1. Wählen Sie eine vorhandene Rolle aus, oder erstellen Sie eine neue Servicerolle, die IAM Access Analyzer Zugriff auf den Bucket in Konto B gewährt (in dem Ihre CloudTrail Logs gespeichert sind).
2. Überprüfen Sie Ihre Amazon-S3-Bucket-Objekteigentümerschaft und die Bucket-Berechtigungsrichtlinie in Konto B, damit IAM Access Analyzer auf Objekte im Bucket zugreifen kann.

Schritt 1: Wählen oder erstellen Sie eine Rolle für kontoübergreifenden Zugriff

- Auf dem Bildschirm Richtlinie generieren ist die Option Vorhandene Rolle verwenden für Sie vorausgewählt, wenn eine Rolle mit den erforderlichen Berechtigungen in Ihrem Konto vorhanden ist. Wählen Sie andernfalls Erstellen und Verwenden einer neuen Servicerolle. Die neue Rolle wird verwendet, um IAM Access Analyzer Zugriff auf Ihre CloudTrail Logs in Konto B zu gewähren.

Schritt 2: Überprüfen oder aktualisieren Sie Ihre Amazon-S3-Bucket-Konfiguration in Konto B

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Bucket-Liste den Namen des Buckets aus, in dem Ihre CloudTrail Trail-Logs gespeichert sind.
3. Wählen Sie die Registerkarte Permissions (Berechtigungen) und gehen Sie zum Abschnitt Object Ownership (Objekteigentümerschaft).

   Verwenden Sie die Bucket-Einstellungen für die Amazon-S3-Objekteigentümerschaft, um die Eigentümerschaft von Objekten zu kontrollieren, die Sie in Ihre Buckets hochladen. Wenn andere Objekte in Ihren Bucket AWS-Konten hochladen, gehören die Objekte standardmäßig dem hochladenden Konto. Um eine Richtlinie zu erstellen, muss der Bucket-Besitzer Eigentümer aller Objekte im Bucket sein. Je nach Anwendungsfall Ihrer ACL müssen Sie möglicherweise die Einstellung für die Object Ownership (Objekteigentümerschaft) für Ihren Bucket ändern. Legen Sie Object Ownership (Objekteigentümerschaft) auf eine der folgenden Optionen fest.

   - Bucket owner enforced (Bucket-Eigentümer erzwungen) (empfohlen)

- Bucket owner preferred (Bucket-Eigentümer bevorzugt)

> **Important**
>
> Um eine Richtlinie erfolgreich zu generieren, müssen die Objekte im Bucket im Besitz des -Bucket-Eigentümers sein. Wenn Sie Bucket owner preferred (Bucket-Besitzer bevorzugt) verwenden, können Sie nur eine Richtlinie für den Zeitraum nach der Änderung der Objekteigentümerschaft erstellen.

Weitere Informationen zur Objekteigentümerschaft in Amazon-S3, finden Sie unter Steuern der Objekteigentümerschaft und Deaktivieren von ACLs für Ihren Bucket im Amazon-S3-Benutzerhandbuch.

4. Fügen Sie Berechtigungen zu Ihrer Amazon S3 Bucket-Richtlinie in Konto B hinzu, um den Zugriff für die Rolle in Konto A zu gewähren.

   Die folgende Beispielrichtlinie erlaubt `ListBucket` und `GetObject` für den Bucket `DOC-EXAMPLE-BUCKET`. Es erlaubt den Zugriff, wenn die Rolle, die auf den Bucket zugreift, zu einem Konto in Ihrer Organisation gehört und einen Namen hat, der mit `AccessAnalyzerMonitorServiceRole` beginnt. Durch die Verwendung `aws:PrincipalArn` als `Condition` im `Resource` Element wird sichergestellt, dass die Rolle nur dann auf Aktivitäten für das Konto zugreifen kann, wenn es zu Konto A gehört. Sie können es `DOC-EXAMPLE-BUCKET` durch Ihren Bucket-Namen, `optional-prefix` durch ein optionales Präfix für den Bucket und `organization-id` durch Ihre Organisations-ID ersetzen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
```

```
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. Wenn Sie Ihre Protokolle mit verschlüsseln AWS KMS, aktualisieren Sie Ihre AWS KMS Schlüsselrichtlinie in dem Konto, in dem Sie die CloudTrail Protokolle speichern, um IAM Access Analyzer Zugriff auf die Verwendung Ihres Schlüssels zu gewähren, wie im folgenden Richtlinienbeispiel gezeigt. Ersetzen Sie `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` durch den ARN für Ihren Trail und `organization-id` durch Ihre Organisations-ID.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
```

```
            "s3.*.amazonaws.com"
          ]
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

Generieren Sie eine Richtlinie auf der Grundlage von CloudTrail Aktivitäten (AWS CLI)

Sie können die folgenden Befehle verwenden, um eine Richtlinie mit AWS CLI zu generieren.

Erstellen einer Richtlinie

- als Accessanalyzer start-policy-generation

Anzeigen einer generierten Richtlinie

- als Zugriffsanalysator get-generated-policy

Stornieren einer Anforderung zur Richtliniengenerierung

- als Zugriffsanalysator cancel-policy-generation

Anzeigen einer Liste von Anforderungen zur Richtliniengenerierung

- als Zugriffsanalysator list-policy-generations

Generieren Sie eine Richtlinie auf der Grundlage von CloudTrail Aktivitäten (AWS API)

Sie können die folgenden Operationen verwenden, um mithilfe der AWS API eine Richtlinie zu generieren.

Erstellen einer Richtlinie

- StartPolicyGenerierung

Anzeigen einer generierten Richtlinie

- GetGeneratedPolitik

Stornieren einer Anforderung zur Richtliniengenerierung

- CancelPolicyGeneration

Anzeigen einer Liste von Anforderungen zur Richtliniengenerierung

- ListPolicyGenerationen

IAM Access Analyzer Services zur Richtliniengenerierung

In der folgenden Tabelle sind die AWS Dienste aufgeführt, für die IAM Access Analyzer Richtlinien mit Informationen auf Aktionsebene generiert. Eine Liste der Aktionen in den einzelnen Diensten finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste in der Service Authorization Reference.

Service	Servicepräfix
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Account Management	Konto
AWS Certificate Manager	acm
Amazon Managed Workflows für Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations

Service	Servicepräfix
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Cost Profiler für Anwendungen	application-cost-profiler
Einblicke in CloudWatch Amazon-Anwendungen	applicationinsights
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	automatische Skalierung
AWS Marketplace	aws-marketplace
AWS Backup	backup
AWS Batch	Batch
Amazon Braket	braket
AWS Budgets	Budgets
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront

Service	Servicepräfix
AWS CloudHSM	cloudhsm
Amazon CloudSearch	clousearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Amazon-Rezensent	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Benachrichtigungen	codestar-notifications
Amazon Cognito-Identität	cognito-identity
Amazon-Cognito-Benutzerpools	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	connect
AWS Cost and Usage Report	cur

Service	Servicepräfix
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
DevOpsAmazon-Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Elastische Amazon DocumentDB-Cluster	docdb-elastic
AWS Directory Service	ds
Amazon-DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache

Service	Servicepräfix
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR auf EKS (EMR Container)	emr-containers
Amazon EMR Serverless	emr-serverless
OpenSearch Amazon-Dienst	es
Amazon EventBridge	Veranstaltungen
Amazon CloudWatch offenbar	evidently
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass

Service	Servicepräfix
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS Identitätsspeicher	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming für Apache Kafka	kafka

Service	Servicepräfix
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	Lambda
Amazon Lex	lex
AWS License Manager Linux-Abonnement-Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
CloudWatch Amazon-Protokolle	Protokolle
Amazon Lookout für Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore

Service	Servicepräfix
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB für Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Migration Hub Strategie-Empfehlungen	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	Organisationen
AWS Panorama	panorama
AWS Performance Insights	pi
EventBridge Amazon-Pfeifen	pipes
Amazon Polly	polly
Amazon Connect Customer Profiles	Profil
Amazon QLDB	qldb

Service	Servicepräfix
AWS Resource Access Manager	ram
AWS Papierkorb	rbin
Amazon Relational Database Service	rds
Amazon-Redshift	redshift
Amazon Redshift-Daten-API	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	Rekognition
AWS Resilience Hub	resiliencehub
AWS Ressourcen Explorer	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Rollen überall	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery-Kontrollen	route53-recovery-control-config
Amazon Route 53 Recovery-Bereitschaft	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	S3
Amazon S3 in Outposts	s3-outposts

Service	Servicepräfix
SageMakerGeospatial-Funktionen von Amazon	sagemaker-geospatial
Savings Plans	savingsplans
EventBridge Amazon-Schemas	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint-SMS- und Sprachnachrichten-Service	sms-voice
AWS Snowball	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm

Service	Servicepräfix
AWS Systems Manager Incident Manager	ssm-incidents
AWS Systems Manager für SAP	ssm-sap
AWS Step Functions	Status
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	Tag (Markierung)
Amazon Textract	textract/
Amazon Timestream	timestream
AWS Telco Network Builder	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	Übertragung
Amazon Translate	translate
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	Workspaces

Service	Servicepräfix
AWS X-Ray	xray

IAM Access Analyzer-Kontingente

IAM Access Analyzer verfügt über die folgenden Kontingente:

Ressource	Standardkontingent	Höchstkontingent
Maximale Anzahl von Analysatoren auf Kontoebene pro Analysatortyp pro AWS-Konto pro Region	1	1
Maximale Anzahl von Analysatoren auf Organisationsebene pro Analysatortyp pro AWS-Konto pro Region	5	20[1]
Maximale Archivierungsregeln pro Analysator	100 Jede Archivregel kann bis zu 20 Werte pro Kriterium enthalten.	1.000[1]
Maximale Anzahl von Zugriffsvorschauen pro Analysator pro Stunde	1.000	1.000
AWS CloudTrail Protokolldateien, die pro Richtliniengenerierung verarbeitet wurden	100 000	100 000
Generieren von Richtlinien	1	1

Ressource	Standardkontingent	Höchstkontingent
AWS CloudTrail Datengröße bei der Richtliniengenerierung	25 GB	25 GB
AWS CloudTrail Zeitraum der Richtliniengenerierung	90 Tage	90 Tage
Richtliniengenerationen pro Tag	Afrika (Kapstadt): 5 Asien-Pazifik (Hongkong): 5 Europa (Mailand): 5 Naher Osten (Bahrain): 5 Alle anderen unterstützten Regionen: 50 **Note** Abgebrochene Richtliniengenerierungsanforderungen gelten für das Tageskontingent.	Afrika (Kapstadt): 5 Asien-Pazifik (Hongkong): 5 Europa (Mailand): 5 Naher Osten (Bahrain): 5 Alle anderen unterstützten Regionen: 50

[1]Einige Kontingente sind mit Hilfe von Service Quotas vom Kunden konfigurierbar.

Fehlersuche bei IAM

Wenn bei der Arbeit mit AWS Identity and Access Management (IAM) der Zugriff verweigert wurde oder andere Schwierigkeiten aufgetreten sind, finden Sie unter den entsprechenden Themen in diesem Abschnitt weitere Informationen.

Themen

Fehlerbehebung bei allgemeinen IAM-Problemen

Nutzen Sie die hier aufgeführten Informationen, um häufige Probleme bei der Arbeit mit AWS Identity and Access Management (IAM) zu diagnostizieren und zu beheben.

Problembereiche

Ich kann mich nicht in meinem AWS -Konto anmelden

Stellen Sie sicher, dass Sie über die richtigen Anmeldeinformationen verfügen und dass Sie die richtige Methode verwenden, um sich anzumelden. Weitere Informationen finden Sie unter Behebung von Anmeldefehlern im AWS-Anmeldung -Benutzerhandbuch.

Ich habe meine Zugriffsschlüssel verloren

Die Zugriffsschlüssel bestehen aus zwei Teilen:

- Die Zugriffsschlüssel-ID. Diese ID ist nicht geheim und kann in der IAM-Konsole dort eingesehen werden, wo Zugriffsschlüssel aufgelistet sind, wie z. B. auf der Übersichtsseite des Benutzers.
- Der geheime Zugriffsschlüssel. Der geheime Zugriffsschlüssel wird bereitgestellt, wenn Sie zum ersten Mal das Zugriffsschlüsselpaar erstellen. Wie bei einem Passwort kann der Schlüssel später nicht mehr abgerufen werden. Wenn Sie Ihren geheimen Zugriffsschlüssel verlieren, müssen Sie ein neues Zugriffsschlüsselpaar erstellen. Wenn Sie bereits über die maximale Anzahl an Zugriffsschlüsseln verfügen, müssen Sie ein vorhandenes Schlüsselpaar löschen, bevor Sie ein anderes erstellen können.

Weitere Informationen finden Sie unter Zurücksetzen verlorener oder vergessener Passwörter oder Zugangsschlüssel für AWS.

Die Richtlinienvariablen funktionieren nicht

- Überprüfen Sie, ob für alle Richtlinien, die Variablen enthalten, die folgende Versionsnummer in der Richtlinie angegeben ist: `"Version": "2012-10-17"`. Ohne die korrekte Versionsnummer werden die Variablen während der Auswertung nicht ersetzt. Stattdessen werden die Variablen wörtlich ausgewertet. Alle Richtlinien, die keine Variablen enthalten, bleiben funktionsfähig, wenn Sie die neuste Versionsnummer angeben.

  Das Richtlinienelement `Version` unterscheidet sich von einer Richtlinienversion. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Andererseits wird eine Richtlinienversion erstellt, wenn Sie in IAM eine benutzerdefinierte, verwaltete Richtlinie bearbeiten. Die vorhandene Richtlinie wird von der geänderten Richtlinie nicht überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. Weitere Informationen zum Richtlinienelement `Version` finden Sie unter IAM-JSON-Richtlinienelemente: Version. Weitere Informationen zu den Richtlinienversionen finden Sie unter the section called “Versioning von IAM-Richtlinien”.

- Überprüfen Sie, ob die Groß- und Kleinschreibung der Richtlinienvariablen richtig ist. Details hierzu finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Änderungen, die ich vornehme, sind nicht immer direkt sichtbar

Als Service, auf den Computer in weltweit angesiedelten Rechenzentren zugreifen, nutzt IAM ein verteiltes Computing-Modell namens Eventual consistency. Jede Änderung, die Sie an IAM (oder anderen AWS Diensten) vornehmen, einschließlich der in der attributebasierten Zugriffskontrolle (ABAC) verwendeten Tags, dauert einige Zeit, bis sie von allen möglichen Endpunkten aus sichtbar wird. Die Verzögerung ergibt sich teilweise aus der Zeit, die erforderlich ist, um die Daten von Server zu Server, von Replikationszone zu Replikationszone und von einer Region der Welt in eine andere zu senden. IAM; verwendet darüber hinaus Zwischenspeicherungen zur Verbesserung der Leistung, doch in einigen Fällen kann dies Zeit hinzufügen. Die Änderung ist möglicherweise erst sichtbar, wenn die Zeit für die vorher zwischengespeicherten Daten abgelaufen ist.

Sie müssen Ihre globalen Anwendungen unter Berücksichtigung dieser potenziellen Verzögerungen konzipieren. Stellen Sie sicher, dass sie wie erwartet funktionieren, und zwar auch dann, wenn eine Änderung an einem Speicherort nicht sofort an einem anderen sichtbar ist. Solche Änderungen umfassen das Erstellen oder Aktualisieren von Benutzern, Gruppen, Rollen und Richtlinien. Wir empfehlen, dass Sie in den kritischen, hochverfügbaren Code-Pfaden Ihrer Anwendung keine solchen IAM-Änderungen vornehmen. Nehmen Sie IAM-Änderungen stattdessen in einer separaten Initialisierungs- oder Einrichtungsroutine vor, die seltener ausgeführt wird. Vergewissern Sie sich auch, dass die Änderungen weitergegeben wurden, bevor die Produktionsarbeitsabläufe davon abhängen.

Weitere Informationen darüber, wie einige andere AWS Dienste davon betroffen sind, finden Sie in den folgenden Ressourcen:

- Amazon DynamoDB: Wie lautet das Konsistenzmodell von Amazon DynamoDB? im DynamoDB – Häufig gestellte Fragen, und Lesekonsistenz im Entwicklerleitfaden für Amazon DynamoDB.
- Amazon EC2: EC2-Eventual Consistency im Amazon EC2 API-Referenz.
- Amazon EMR: Sicherstellung der Konsistenz bei der Verwendung von Amazon S3 und Amazon Elastic MapReduce für ETL-Workflows im AWS Big Data-Blog
- Amazon Redshift: Verwalten der Datenkonsistenz im Amazon Redshift Developerhandbuch
- Amazon S3: Amazon-S3-Datenkonsistenzmodell im Benutzerhandbuch für Amazon Simple Storage Service

Ich bin nicht berechtigt, Folgendes auszuführen: iam: MFADevice DeleteVirtual

Möglicherweise erhalten Sie die folgende Fehlermeldung, wenn Sie versuchen, ein virtuelles MFA-Gerät für sich selbst oder andere zuzuweisen oder zu entfernen:

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to perform:
 iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego with an
 explicit deny
```

Dies kann passieren, wenn jemand zuvor damit begonnen hat, ein virtuelles MFA-Gerät zu einem Benutzer in der IAM-Konsole zuzuweisen und den Vorgang abgebrochen hat. Dadurch wird ein MFA-Gerät für den Benutzer in IAM erstellt, aber diesem niemals zugeordnet. Sie müssen das vorhandene virtuelle MFA-Gerät löschen, bevor Sie ein neues virtuelles MFA-Gerät mit dem gleichen Gerätenamen erstellen können.

Zur Behebung des Problems sollte ein Administrator keine Richtlinienberechtigungen bearbeiten. Stattdessen muss der Administrator die AWS API AWS CLI oder verwenden, um das vorhandene, aber nicht zugewiesene virtuelle MFA-Gerät zu löschen.

So löschen Sie ein vorhandenes, aber nicht zugeordnetes virtuelles MFA-Gerät

1. Zeigen Sie die virtuellen MFA-Geräte in Ihrem Konto an.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API: `ListVirtualMFADevices`
2. Suchen Sie in der Antwort den ARN des virtuellen MFA-Geräts für den Benutzer, für den Sie eine Korrektur vornehmen möchten.
3. Löschen Sie das virtuelle MFA-Gerät.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - AWS API: `DeleteVirtualMFADevice`

Wie erstelle ich sicher IAM-Benutzer?

Wenn Sie Mitarbeiter haben, auf die Sie Zugriff benötigen AWS, können Sie IAM-Benutzer erstellen oder IAM Identity Center für die Authentifizierung verwenden. Wenn Sie IAM verwenden, AWS empfiehlt Ihnen, einen IAM-Benutzer zu erstellen und die Anmeldeinformationen dem Mitarbeiter

auf sichere Weise mitzuteilen. Wenn Sie sich nicht physisch neben Ihrem Mitarbeiter befinden, verwenden Sie einen sicheren Workflow, um den Mitarbeitern Anmeldeinformationen zu übermitteln.

Verwenden Sie den folgenden Workflow, um sicher einen neuen Benutzer in IAM zu erstellen:

1. Erstellen Sie einen neuen Benutzer mit der AWS Management Console. Wählen Sie, ob Sie AWS Management Console den Zugriff mit einem automatisch generierten Passwort gewähren möchten. Aktivieren Sie ggf. das Kontrollkästchen Benutzer müssen bei der nächsten Anmeldung ein neues Passwort erstellen. Fügen Sie dem Benutzer erst dann eine Berechtigungsrichtlinie hinzu, nachdem er sein Passwort geändert hat.
2. Nachdem der Benutzer hinzugefügt wurde, kopieren Sie die Anmelde-URL, den Benutzernamen und das Passwort für den neuen Benutzer. Um das Passwort anzuzeigen, wählen Sie Anzeigen.
3. Senden Sie das Passwort an Ihren Mitarbeiter mit einer sicheren Kommunikationsmethode in Ihrem Unternehmen, wie E-Mail, Chat oder einem Ticketsystem. Geben Sie Ihren Benutzern separat den Link zur IAM-Benutzerkonsole und ihren Benutzernamen an. Sagen Sie dem Mitarbeiter, dass er bestätigen soll, dass er sich erfolgreich anmelden kann, bevor Sie ihm Berechtigungen erteilen.
4. Nachdem der Mitarbeiter dies bestätigt hat, fügen Sie die erforderlichen Berechtigungen hinzu. Fügen Sie als bewährte Methode für die Sicherheit eine Richtlinie hinzu, die erfordert, dass sich der Benutzer mit MFA authentifiziert, um seine Anmeldeinformationen zu verwalten. Eine Beispielrichtlinie finden Sie unter AWS: Ermöglicht MFA-authentifizierten IAM-Benutzern, ihre eigenen Anmeldeinformationen auf der Seite Sicherheitsanmeldedaten zu verwalten.

Weitere Ressourcen

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

- AWS CloudTrail Benutzerhandbuch — Wird verwendet AWS CloudTrail , um den Verlauf von API-Aufrufen nachzuverfolgen AWS und diese Informationen in Protokolldateien zu speichern. Auf diese Weise können Sie einfacher bestimmen, welche Benutzer und Konten auf Ressourcen in Ihrem Konto zugegriffen haben, wann die Aufrufe erfolgten, welche Aktionen angefordert wurden, und vieles mehr. Weitere Informationen finden Sie unter Protokollierung von IAM- und AWS STS API-Aufrufen mit AWS CloudTrail.
- AWS Knowledge Center — Hier finden Sie häufig gestellte Fragen und Links zu anderen Ressourcen, die Ihnen bei der Behebung von Problemen helfen.
- AWS Support Center — Holen Sie sich technischen Support.

- AWS Premium Support Center — Holen Sie sich erstklassigen technischen Support.

Fehlerbehebung von Meldungen aufgrund Zugriffsverweigerung

Fehler „Zugriff verweigert“ treten auf, wenn eine Autorisierungsanfrage AWS explizit oder implizit verweigert wird. Eine ausdrückliche Ablehnung liegt vor, wenn eine Richtlinie eine `Deny` Anweisung für die spezifische Aktion enthält. AWS Eine implizite Verweigerung tritt auf, wenn es keine entsprechende `Deny`-Anweisung, aber auch keine anwendbare `Allow`-Anweisung gibt. Da eine IAM-Richtlinie einen IAM-Prinzipal standardmäßig verweigert, muss die Richtlinie dem Prinzipal ausdrücklich erlauben, eine Aktion auszuführen. Andernfalls verweigert die Richtlinie implizit den Zugriff. Weitere Informationen finden Sie unter Der Unterschied zwischen expliziten und impliziten Zugriffsverweigerungen.

Wenn mehrere Richtlinien desselben Richtlinientyps eine Autorisierungsanforderung verweigern, dann gibt AWS die Nummer in der Zugriffsverweigerungs-Fehlermeldung nicht an. Wenn mehrere Richtlinientypen eine Autorisierungsanfrage ablehnen AWS , wird nur einer dieser Richtlinientypen in die Fehlermeldung aufgenommen.

> **Important**
>
> Haben Sie Probleme bei der Anmeldung AWS? Stellen Sie sicher, dass Sie sich auf der richtigen AWS -Anmeldeseite für Ihren Benutzertyp befinden. Wenn Sie der Root-Benutzer des AWS-Kontos (Kontoinhaber) sind, können Sie sich AWS mit den Anmeldeinformationen anmelden, die Sie bei der Erstellung des eingerichtet haben AWS-Konto. Wenn Sie ein IAM-Benutzer sind, kann Ihr Kontoadministrator Ihnen die Anmeldeinformationen bereitstellen, mit denen Sie sich bei AWS anmelden können. Wenn Sie Support anfordern müssen, verwenden Sie nicht den Feedback-Link auf dieser Seite, da das Formular beim AWS Dokumentationsteam eingegangen ist, nicht AWS Support. Wählen Sie stattdessen auf der Seite „Kontaktieren Sie uns“ die Option Immer noch nicht in Ihr AWS Konto einloggen und wählen Sie dann eine der verfügbaren Support-Optionen aus.

Ich erhalte die Meldung „Zugriff verweigert“, wenn ich eine Anfrage an einen AWS Dienst stelle

- Überprüfen Sie, ob die Fehlermeldung den Typ der Richtlinie enthält, die für die Verweigerung des Zugriffs verantwortlich ist. Wenn der Fehler beispielsweise erwähnt, dass der Zugriff aufgrund

einer Dienststeuerungsrichtlinie (Service Control Policy, SCP) verweigert wird, können Sie sich auf die Problembehandlung von SCP-Problemen konzentrieren. Wenn Sie den Richtlinientyp kennen, können Sie auch für die bestimmte Aktion in Richtlinien dieses Richtlinientyps nach einer Deny-Anweisung oder einem fehlenden Zulassungsfehler suchen. Wenn in der Fehlermeldung der Richtlinientyp, der für die Verweigerung des Zugriffs verantwortlich ist, nicht erwähnt wird, verwenden Sie die restlichen Richtlinien in diesem Abschnitt, um weitere Probleme zu beheben.

- Stellen Sie sicher, dass Sie über die identitätsbasierten Richtlinienberechtigungen zum Aufrufen der Aktion und Ressource verfügen, die Sie angefordert haben. Wenn Bedingungen definiert sind, müssen Sie diese Bedingungen beim Senden der Anforderung ebenfalls erfüllen. Weitere Informationen zum Anzeigen oder Ändern von Richtlinien für einen IAM-Benutzer, einer -Gruppe oder -Rolle finden Sie unter Verwalten von IAM-Richtlinien.
- Wenn eine Meldung AWS Management Console zurückgegeben wird, dass Sie nicht berechtigt sind, eine Aktion auszuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen oder Anmelde-Link zur Verfügung gestellt.

  Der folgende Beispielfehler tritt auf, wenn der `mateojackson`-IAM-Benutzer versucht, die Konsole zum Anzeigen von Details zu einem *`my-example-widget`* zu verwenden, jedoch nicht über `widgets:`*`GetWidget`*-Berechtigungen verfügt.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion *`my-example-widget`* auf die Ressource `widgets:`*`GetWidget`* zugreifen zu können.
- Versuchen Sie, auf einen Service zuzugreifen, der Ressourcenbasierte Richtlinien wie Amazon S3, Amazon SNS oder Amazon SQS unterstützt? Wenn dies der Fall ist, stellen Sie sicher, dass Sie in der Ressourcenrichtlinie als Auftraggeber aufgeführt sind und Ihnen Zugriff erteilt wurde. Wenn Sie eine Anfrage an einen Service innerhalb Ihres Kontos machen, können Ihnen entweder Ihre identitätsbasierte Richtlinien oder die ressourcenbasierten Richtlinien Berechtigung erteilen. Wenn Sie eine Anfrage an einen Service in einem anderen Kontos machen, dann müssen Ihnen sowohl Ihre identitätsbasierte Richtlinien als auch die ressourcenbasierten Richtlinien die Berechtigung erteilen. Informationen dazu, welche Services ressourcenbasierte Richtlinien unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

- Wenn Ihre Richtlinie eine Bedingung mit einem Schlüssel-Wert-Paar umfasst, überprüfen Sie sie sorgfältig. Beispiele hierfür sind der `aws:RequestTag/tag-key`globale Bedingungsschlüssel AWS KMS `kms:EncryptionContext:encryption_context_key`, der und der `ResourceTag/tag-key` Bedingungsschlüssel, die von mehreren Diensten unterstützt werden. Stellen Sie sicher, dass der Schlüsselname nicht mit mehreren Ergebnissen übereinstimmt. Da bei Bedingungsschlüsselnamen die Groß-/Kleinschreibung nicht berücksichtigt wird, gibt eine Bedingung, die nach einem Schlüssel namens `foo` sucht, `foo`, `Foo` oder `F00`. Wenn Ihre Anforderung mehrere Schlüssel-Wert-Paare mit Schlüsselnamen enthält, die sich nur durch die Groß- und Kleinschreibung unterscheiden, wird der Zugriff möglicherweise unerwartet verweigert. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Condition.
- Wenn Sie eine Berechtigungsgrenze haben, überprüfen Sie, ob die für die Berechtigungsgrenze verwendete Richtlinie Ihre Anfrage zulässt. Wenn Ihre identitätsbasierten Richtlinien die Anforderung zulassen, Ihre Berechtigungsgrenze dies jedoch nicht tut, wird die Anforderung abgelehnt. Eine Berechtigungsgrenze bestimmt die maximalen Berechtigungen, die ein IAM-Auftraggeber (Benutzer oder Rolle) haben kann. Ressourcenbasierte Richtlinien werden nicht von Berechtigungsgrenzen beschränkt. Berechtigungsgrenzen sind nicht allgemein üblich. Weitere Informationen darüber, wie Richtlinien AWS bewertet werden, finden Sie unterAuswertungslogik für Richtlinien.
- Wenn Sie API-Anfragen manuell signieren (ohne Verwendung von AWS -SDKs), stellen Sie sicher, dass Sie die Anfrage korrekt signiert haben.

Ich erhalte eine Meldung, dass der Zugriff verweigert wird, wenn ich eine Anfrage mit temporären Sicherheitsanmeldeinformationen stelle

- Stellen Sie zunächst sicher, dass der Zugriff nicht aus einem Grund verweigert wird, der nicht mit Ihren temporären Anmeldeinformationen in Verbindung steht. Weitere Informationen finden Sie unter Ich erhalte die Meldung „Zugriff verweigert“, wenn ich eine Anfrage an einen AWS Dienst stelle.
- Überprüfen Sie, dass der Service temporäre Sicherheitsanmeldeinformationen zulässt. Informationen dazu finden Sie unter AWS Dienste, die mit IAM funktionieren.
- Stellen Sie sicher, dass Ihre Anfragen korrekt signiert sind und die Anfrage richtig aufgebaut ist. Weitere Informationen finden Sie in der Dokumentation zum Toolkit oder unter Verwenden temporärer Anmeldeinformationen mit AWS -Ressourcen.
- Stellen Sie sicher, dass die temporären Sicherheitsanmeldeinformationen nicht abgelaufen sind. Weitere Informationen finden Sie unter Temporäre IAM Sicherheitsanmeldeinformationen.

- Überprüfen Sie, ob der IAM-Benutzer oder die IAM-Rolle über die richtigen Berechtigungen verfügt. Berechtigungen für temporäre Sicherheitsanmeldeinformationen werden von einem IAM-Benutzer oder einer Rolle abgeleitet. Dies hat zur Folge, dass die Berechtigungen auf diejenigen beschränkt sind, die der Rolle gewährt werden, deren temporäre Anmeldeinformationen Sie angenommen haben. Weitere Informationen zum Festlegen von Berechtigungen in temporären Sicherheitsanmeldeinformationen finden Sie unter Steuern von Berechtigungen für temporäre Sicherheitsanmeldeinformationen.
- Wenn Sie eine Rolle übernommen haben, wird Ihre Rollensitzung möglicherweise durch Sitzungsrichtlinien eingeschränkt. Wenn Sie temporäre Sicherheitsanmeldeinformationen programmgesteuert anfordernAWS STS, können Sie optional Inline-Richtlinien oder verwaltete Sitzungsrichtlinien übergeben. Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie programmgesteuert eine Sitzung mit temporären Anmeldeinformationen für eine Rolle erstellen. Sie können ein einzelnes JSON-Inline-Sitzungsrichtliniendokument mit dem `Policy`-Parameter übergeben. Sie können mit dem Parameter `PolicyArns` bis zu 10 verwaltete Sitzungsrichtlinien angeben. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Es kann auch sein, dass, wenn Ihr Administrator oder ein benutzerdefiniertes Programm Ihnen temporäre Anmeldeinformationen bereitstellt, bereits eine Sitzungsrichtlinie zum Einschränken Ihres Zugriffs enthalten ist.
- Wenn Sie ein Verbundbenutzer sind, wird Ihre Sitzung möglicherweise durch Sitzungsrichtlinien beschränkt. Sie werden ein Verbundbenutzer, indem Sie sich AWS als IAM-Benutzer anmelden und dann ein Verbund-Token anfordern. Weitere Informationen zu Verbundbenutzern finden Sie unter GetFederationToken - Verbund durch einen benutzerdefinierten Identity Broker. Wenn Sie oder Ihr Identity Broker während der Anforderung eines Verbund-Tokens Sitzungsrichtlinien übergeben haben, wird Ihre Sitzung durch diese Richtlinien beschränkt. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des IAM-Benutzers basierenden Richtlinien und der Sitzungsrichtlinien. Weitere Informationen zu Sitzungsrichtlinien finden Sie unter Sitzungsrichtlinien.
- Wenn Sie auf eine Ressource mit einer ressourcenbasierten Richtlinie über eine Rolle zugreifen, überprüfen Sie, ob die Richtlinie der Rolle Berechtigungen erteilt. Die folgende Richtlinie beispielsweise erteilt `MyRole` im Konto `111122223333` die Zugriffsberechtigung auf `MyBucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
```

```
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::MyBucket/*"]
  }]
}
```

Beispiele für Zugriffsverweigerungs-Fehlermeldungen

Die meisten Zugriffsverweigerungs-Fehlermeldungen haben das Format `User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. In diesem Beispiel ist der *`Benutzer`* der Amazon-Ressourcenname (ARN), der keinen Zugriff erhält, die *`Aktion`* ist die Serviceaktion, die die Richtlinie verweigert, und die *`Ressource`* ist der ARN der Ressource, auf der die Richtlinie wirkt. Das Feld *`context`* (Kontext) stellt zusätzlichen Kontext über den Richtlinientyp dar, der erklärt, warum die Richtlinie den Zugriff verweigert.

Wenn eine Richtlinie den Zugriff explizit verweigert, weil die Richtlinie eine `Deny` Anweisung enthält, wird AWS der Ausdruck `with an explicit deny in a` *`type`* `policy` in die Fehlermeldung „Zugriff verweigert“ aufgenommen. Wenn die Richtlinie implizit den Zugriff verweigert, wird der Ausdruck `because no` *`type`* `policy allows the` *`action`* `action` in AWS die Fehlermeldung „Zugriff verweigert“ aufgenommen.

> **Note**
>
> Einige AWS Dienste unterstützen dieses Format für die Fehlermeldung „Zugriff verweigert“ nicht. Der Inhalt der Fehlermeldungen bei verweigertem Zugriff kann je nach Service, der die Autorisierungsanforderung stellt, variieren.

Die folgenden Beispiele zeigen das Format für verschiedene Arten von Zugriffsverweigerungs-Fehlermeldungen.

Zugriffsverweigerung aufgrund einer Service-Kontrollrichtlinie – implizite Verweigerung

1. Überprüfen Sie, ob in Ihren Service-Kontrollrichtlinien (SCPs) eine `Allow`-Anweisung für die Aktion fehlt. Für das folgende Beispiel lautet die Aktion `codecommit:ListRepositories`.
2. Aktualisieren Sie Ihre SCP, indem Sie die `Allow`-Anweisung hinzufügen. Weitere Informationen finden Sie unter -Over-the-Air-Updates im AWS Organizations -Leitfaden.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no service control policy allows the
 codecommit:ListRespositories action
```

Zugriffsverweigerung aufgrund einer Service-Kontrollrichtlinie – explizite Verweigerung

1. Überprüfen Sie, ob in Ihren Service-Kontrollrichtlinien (SCPs) eine `Deny`-Anweisung für die Aktion vorhanden ist. Für das folgende Beispiel lautet die Aktion `codecommit:ListRepositories`.
2. Aktualisieren Sie Ihren SCP, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter -Over-the-Air-Updates im AWS Organizations -Leitfaden.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories with an explicit deny in a service control policy
```

Zugriff aufgrund einer VPC-Endpunktrichtlinie verweigert – implizite Ablehnung

1. Überprüfen Sie in Ihren Virtual Private Cloud (VPC)-Endpunktrichtlinien, ob eine `Allow`-Anweisung für die Aktion fehlt. Für das folgende Beispiel lautet die Aktion `codecommit:ListRepositories`.
2. Aktualisieren Sie Ihre VPC-Endpunktrichtlinie, indem Sie die `Allow`-Anweisung hinzufügen. Weitere Informationen finden Sie unter Aktualisieren einer VPC-Endpunktrichtlinie im AWS PrivateLink -Handbuch.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no VPC endpoint policy allows the
 codecommit:ListRepositories action
```

Zugriff aufgrund einer VPC-Endpunktrichtlinie verweigert – explizite Ablehnung

1. Überprüfen Sie, ob in Ihren Virtual Private Cloud (VPC)-Endpunktrichtlinien eine explizite `Deny`-Anweisung für die Aktion vorhanden ist. Für das folgende Beispiel lautet die Aktion `codedeploy:ListDeployments`.
2. Aktualisieren Sie Ihre VPC-Endpunktrichtlinie, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter Aktualisieren einer VPC-Endpunktrichtlinie im AWS PrivateLink -Handbuch.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a VPC endpoint policy
```

Zugriff aufgrund einer Berechtigungsgrenze verweigert – implizite Ablehnung

1. Überprüfen Sie, ob in Ihrer Berechtigungsgrenze eine `Allow`-Anweisung für die Aktion fehlt. Für das folgende Beispiel lautet die Aktion `codedeploy:ListDeployments`.
2. Aktualisieren Sie Ihre Berechtigungsgrenze, indem Sie die `Allow`-Anweisung zu Ihrer IAM-Richtlinie hinzufügen. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* because no permissions boundary allows the
 codedeploy:ListDeployments action
```

Zugriff aufgrund einer Berechtigungsgrenze verweigert – explizite Ablehnung

1. Überprüfen Sie, ob in Ihren Berechtigungsgrenzen eine explizite `Deny`-Anweisung für die Aktion vorhanden ist. Für das folgende Beispiel lautet die Aktion `sagemaker:ListModels`.
2. Aktualisieren Sie Ihre Berechtigungsgrenze, indem Sie die `Deny`-Anweisung aus Ihrer IAM-Richtlinie entfernen. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sagemaker:ListModels with an explicit deny in a permissions boundary
```

Zugriff aufgrund von Sitzungsrichtlinien verweigert – implizite Ablehnung

1. Überprüfen Sie, ob in Ihren Sitzungsrichtlinien eine `Allow`-Anweisung für die Aktion fehlt. Für das folgende Beispiel lautet die Aktion `codecommit:ListRepositories`.
2. Aktualisieren Sie Ihre Sitzungsrichtlinie, indem Sie die `Allow`-Anweisung hinzufügen. Weitere Informationen finden Sie unter Sitzungsrichtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no session policy allows the
 codecommit:ListRepositories action
```

Zugriff aufgrund von Sitzungsrichtlinien verweigert – explizite Ablehnung

1. Überprüfen Sie, ob in Ihren Sitzungsrichtlinien eine explizite `Deny`-Anweisung für die Aktion vorhanden ist. Für das folgende Beispiel lautet die Aktion `codedeploy:ListDeployments`.
2. Aktualisieren Sie Ihre Sitzungsrichtlinie, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter Sitzungsrichtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a sessions policy
```

Zugriff aufgrund ressourcenbasierter Richtlinien verweigert – implizite Ablehnung

1. Überprüfen Sie, ob in Ihrer ressourcenbasierten Richtlinie eine `Allow`-Anweisung für die Aktion fehlt. Für das folgende Beispiel lautet die Aktion `secretsmanager:GetSecretValue`.
2. Aktualisieren Sie Ihre Richtlinie, indem Sie die `Allow`-Anweisung hinzufügen. Weitere Informationen finden Sie unter Ressourcenbasierte Richtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue because no resource-based policy allows the
 secretsmanager:GetSecretValue action
```

Zugriff aufgrund ressourcenbasierter Richtlinien verweigert – explizite Ablehnung

1. Suchen Sie nach einer expliziten `Deny`-Anweisung für die Aktion in Ihrer ressourcenbasierten Richtlinie. Für das folgende Beispiel lautet die Aktion `secretsmanager:GetSecretValue`.
2. Aktualisieren Sie Ihre Richtlinie, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter Ressourcenbasierte Richtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
```

```
secretsmanager:GetSecretValue on resource: arn:aws:secretsmanager:us-
east-1:123456789012:secret:* with an explicit deny in a resource-based policy
```

Zugriff aufgrund von Rollenvertrauensrichtlinien verweigert – implizite Verweigerung

1. Überprüfen Sie, ob in Ihrer Rollenvertrauensrichtlinien eine `Allow`-Anweisung für die Aktion fehlt. Für das folgende Beispiel lautet die Aktion `sts:AssumeRole`.
2. Aktualisieren Sie Ihre Richtlinie, indem Sie die `Allow`-Anweisung hinzufügen. Weitere Informationen finden Sie unter Ressourcenbasierte Richtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
sts:AssumeRole because no role trust policy allows the sts:AssumeRole action
```

Zugriff aufgrund von Rollenvertrauensrichtlinien verweigert – explizite Ablehnung

1. Überprüfen Sie, ob in Ihrer Rollenvertrauensrichtlinie eine explizite `Deny`-Anweisung für die Aktion enthalten ist. Für das folgende Beispiel lautet die Aktion `sts:AssumeRole`.
2. Aktualisieren Sie Ihre Richtlinie, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter Ressourcenbasierte Richtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sts:AssumeRole with an explicit deny in the role trust policy
```

Zugriff aufgrund identitätsbasierter Richtlinien verweigert – implizite Verweigerung

1. Überprüfen Sie ob in identitätsbasierten Richtlinien, die der Identität angefügt sind, eine `Allow`-Anweisung für die Aktion fehlt. Im folgenden Beispiel ist die Aktion `codecommit:ListRepositories` dem Benutzer `JohnDoe` zugeordnet.
2. Aktualisieren Sie Ihre Richtlinie, indem Sie die `Allow`-Anweisung hinzufügen. Weitere Informationen finden Sie unter Identitätsbasierte Richtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no identity-based policy allows the
 codecommit:ListRepositories action
```

Zugriff aufgrund identitätsbasierter Richtlinien verweigert – explizite Ablehnung

1. Überprüfen Sie, ob in identitätsbasierten Richtlinien, die der Identität angefügt sind, eine explizite `Deny`-Anweisung für die Aktion vorhanden ist. Im folgenden Beispiel ist die Aktion `codedeploy:ListDeployments` dem Benutzer `JohnDoe` zugeordnet.
2. Aktualisieren Sie Ihre Richtlinie, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter Identitätsbasierte Richtlinien und Bearbeiten von IAM-Richtlinien.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in an identity-based policy
```

Zugriffsverweigerung, wenn eine VPC-Anforderung aufgrund einer anderen Richtlinie fehlschlägt

1. Überprüfen Sie, ob in Ihren Service Control Policies (SCPs) eine explizite `Deny`-Anweisung für die Aktion vorhanden ist. Für das folgende Beispiel lautet die Aktion `SNS:Publish`.
2. Aktualisieren Sie Ihren SCP, indem Sie die `Deny`-Anweisung entfernen. Weitere Informationen finden Sie unter -Over-the-Air-Updates im AWS IAM Identity Center -Leitfaden.

```
User: arn:aws:sts::111122223333:assumed-role/role-name/role-session-name is not
 authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-1:444455556666:role-name-2
with an explicit deny in a VPC endpoint policy transitively through a service control
 policy
```

Fehlerbehebung bei IAM-Richtlinien

Eine Richtlinie ist eine Entität, AWS die, wenn sie an eine Identität oder Ressource angehängt wird, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Principal, z. B. ein Benutzer, eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Richtlinien werden in Form von JSON-Dokumenten gespeichert, die AWS als identitätsbasierte Richtlinien an Prinzipale oder als ressourcenbasierte Richtlinien an Ressourcen angehängt werden. Sie können eine identitätsbasierte Richtlinie an einen Auftraggeber (oder eine Identität) anfügen – z. B. eine IAM-Gruppe, einen Benutzer oder eine Rolle. Zu identitätsbasierten Richtlinien gehören verwaltete AWS -Richtlinien, kundenverwaltete

Richtlinien und eingebundene Richtlinien. Sie können vom Kunden verwaltete Richtlinien in den AWS Management Console Editor-Optionen Visual und JSON erstellen und bearbeiten. Wenn Sie eine Richtlinie in der anzeigen AWS Management Console, können Sie eine Zusammenfassung der Berechtigungen sehen, die durch diese Richtlinie gewährt werden. Sie können den visuellen Editor und die Richtlinienübersicht zur Diagnose und Behebung allgemeiner Fehler nutzen, die beim Verwalten von IAM-Richtlinien auftreten können.

Beachten Sie, dass alle IAM-Richtlinien mithilfe einer Syntax gespeichert werden, die mit den Regeln der JavaScript Object Notation (JSON) beginnt. Sie müssen diese Syntax nicht verstehen, um Richtlinien erstellen und verwalten zu können. Sie können eine Richtlinie mithilfe des visuellen Editors in der AWS Management Console erstellen und bearbeiten. Weitere Informationen zur JSON-Syntax in IAM-Richtlinien finden Sie unter Grammatik der IAM-JSON-Richtliniensprache .

Fehlerbehebung bei IAM-Richtlinien

- Fehlerbehebung mit dem visuellen Editor
  - Umstrukturierung einer Richtlinie
  - Auswählen eines Ressourcen-ARN im visuellen Editor
  - Verweigern von Berechtigungen im visuellen Editor
  - Angeben mehrerer Services im visuellen Editor
  - Reduzieren der Größe Ihrer Richtlinie im visuellen Editor
  - Beheben von Problemen aufgrund nicht erkannter Services, Aktionen oder Ressourcentypen im visuellen Editor
- Problembehandlung unter Verwendung von Richtlinienübersichten
  - Übersicht fehlender Richtlinien
  - Die Richtlinienübersicht umfasst nicht erkannte Services, Aktionen oder Ressourcentypen.
  - Der Service unterstützt keine IAM-Richtlinienübersichten
  - Meine Richtlinie erteilt nicht die erwarteten Berechtigungen
- Fehlerbehebung bei der Richtlinienverwaltung
  - Anfügen oder Entfernen einer Richtlinie zu/von einem IAM-Konto
  - Ändern von Richtlinien für Ihre IAM-Identitäten basierend auf ihrer Aktivität
- Fehlerbehebung bei JSON-Richtliniendokumenten
  - Validieren Ihrer Richtlinien
  - Ich habe keine Berechtigungen für die Richtlinienvalidierung im JSON-Editor

- Mehrere JSON-Richtlinienobjekte
- Mehrere JSON-Anweisungselemente
- Mehrere Effect-, Action- oder Resource-Elemente in einem JSON-Anweisungselement
- Fehlendes JSON-Versionselement

Fehlerbehebung mit dem visuellen Editor

Wenn Sie eine vom Kunden verwaltete Richtlinie erstellen oder ändern, können Sie die Informationen im Visual-Editor nutzen, um Fehler in Ihrer Richtlinie zu beheben. Ein Beispiel für die Verwendung des visuellen Editors zum Erstellen einer Richtlinie finden Sie unter the section called “Steuern des Zugriffs auf Identitäten”.

Umstrukturierung einer Richtlinie

Wenn Sie eine Richtlinie erstellen, AWS validiert, verarbeitet und transformiert sie, bevor Sie sie speichern. Wenn die Richtlinie als Antwort auf eine Benutzerabfrage AWS zurückgibt oder sie in der Konsole anzeigt, wird die Richtlinie wieder in ein AWS für Menschen lesbares Format umgewandelt, ohne die durch die Richtlinie gewährten Berechtigungen zu ändern. Dies kann zu Abweichungen führen zwischen dem, was Sie im visuellen Editor oder auf der Registerkarte JSON sehen. Es können Berechtigungsblöcke des visuellen Editors hinzugefügt, entfernt und neu organisiert worden sein. Zudem wurde der Inhalt innerhalb eines Blocks möglicherweise optimiert. Auf der Registerkarte JSON können unbedeutende Leerzeichen entfernt und Elemente innerhalb von JSON-Zuordnungen neu sortiert werden. Darüber hinaus können AWS-Konto IDs innerhalb der Hauptelemente durch den ARN des ersetzt werden Root-Benutzer des AWS-Kontos. Aufgrund dieser möglichen Änderungen sollten Sie JSON-Richtliniendokumente nicht als Zeichenfolgen vergleichen.

Wenn Sie in der eine vom Kunden verwaltete Richtlinie erstellen AWS Management Console, können Sie wählen, ob Sie ausschließlich im JSON-Editor arbeiten möchten. Wenn Sie nie Änderungen im Visual-Editor durchführen und im JSON-Editor Next (Weiter) auswählen, ist die Wahrscheinlichkeit, dass die Richtlinie umstrukturiert wird, geringer. Wenn Sie jedoch eine Richtlinie einrichten und den Visual-Editor nutzen, um Änderungen durchzuführen, oder wenn Sie im Visual-Editor die Option Next (Weiter) auswählen, kann es sein, dass IAM die Richtlinie umstrukturiert, um dessen Anzeige im visuellen Editor zu optimieren.

Diese Umstrukturierung erfolgt nur in der Bearbeitungssitzung und wird nicht automatisch gespeichert.

Wenn Ihre Richtlinie während der Bearbeitungssitzung umstrukturiert wird, legt IAM basierend auf den folgenden Situationen fest, ob diese neue Version gespeichert wird:

Diese Editor-Option verwenden	Wenn Sie Ihre Richtlinie bearbeiten	Wählen Sie dann in dieser Registerkarte Next (Weiter) aus	Wenn Sie Save changes (Änderungen speichern) auswählen
Visual	Bearbeitet	Visual	Die Richtlinie wird neu strukturiert
Visual	Bearbeitet	JSON	Die Richtlinie wird neu strukturiert
Visual	Nicht bearbeitet	Visual	Die Richtlinie wird neu strukturiert
JSON	Bearbeitet	Visual	Die Richtlinie wird neu strukturiert
JSON	Bearbeitet	JSON	Die Richtlinienstruktur bleibt unverändert
JSON	Nicht bearbeitet	JSON	Die Richtlinienstruktur bleibt unverändert

IAM strukturiert möglicherweise komplexe Richtlinien oder Richtlinien mit Berechtigungsblöcken oder Anweisungen um, die mehrere Services, Ressourcentypen oder Bedingungsschlüssel zulassen.

Auswählen eines Ressourcen-ARN im visuellen Editor

Wenn Sie mit dem visuellen Editor eine Richtlinie erstellen oder bearbeiten, müssen Sie zunächst einen Service und anschließend Aktionen dieses Services auswählen. Unterstützen die von Ihnen ausgewählten Services und Aktionen die Auswahl bestimmter Ressourcen, werden im visuellen Editor die unterstützten Ressourcentypen aufgeführt. Sie können Add ARN (ARN hinzufügen) auswählen, um Details zu Ihren Ressourcen bereitzustellen. Zum Hinzufügen eines ARNs für einen Ressourcentyp können Sie aus folgenden Optionen auswählen.

- Use the ARN builder – Je nach Ressourcentyp sehen Sie möglicherweise unterschiedliche Felder für die ARN-Generierung. Sie können auch Any (Alle) auswählen, um Berechtigungen für die

einzelnen Werte für die angegebene Einstellung bereitzustellen. Wenn Sie z. B. die Amazon EC2-Zugriffsebenengruppe Lesen ausgewählt haben, unterstützen die Aktionen in Ihrer Richtlinie den `instance`-Ressourcentyp. Sie müssen die Region, das Konto und die InstanceIdWerte für Ihre Ressource angeben. Wenn Sie Ihre Konto-ID angeben, aber Any (Alle) für die Region und die Instance-ID auswählen, dann gewährt die Richtlinie Berechtigungen für alle Instances in Ihrem Konto.

- ARN eingeben oder einfügen – Sie können Ressourcen anhand ihres Amazon-Ressourcennamens (ARN) angeben. Sie können einen Platzhalter (*****) in einem beliebigen Feld des ARNs (zwischen zwei Doppelpunkten) einschließen. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Resource.

Verweigern von Berechtigungen im visuellen Editor

Standardmäßig lässt die Richtlinie, die Sie mit dem visuellen Editor erstellen, die Aktionen zu, die Sie auswählen. Um die ausgewählten Aktionen stattdessen zu verweigern, wählen Sie Switch to deny permissions (Zu Berechtigungen verweigern wechseln). Da Anforderungen standardmäßig verweigert werden, empfehlen wir, dass Sie im Sinne bewährter Methoden nur für jene Aktionen und Ressourcen Berechtigungen erteilen, für die ein Benutzer Zugriff benötigt. Sie sollten nur dann eine Anweisung erstellen, um Berechtigungen zu verweigern, wenn Sie eine von einer anderen Anweisung oder Richtlinie erteilte Berechtigung separat überschreiben möchten. Wir empfehlen, die Anzahl der Verweigerungsberechtigungen auf ein Minimum zu beschränken, da diese die Fehlerbehebung bei Berechtigungen erschweren. Weitere Informationen dazu, wie IAM die Richtlinienlogik auswertet, finden Sie unter Auswertungslogik für Richtlinien.

> **Note**
>
> Standardmäßig Root-Benutzer des AWS-Kontos hat nur der Zugriff auf alle Ressourcen in diesem Konto. Wenn Sie nicht als Stammbenutzer angemeldet sind, müssen Sie über Berechtigungen verfügen, die im Rahmen einer Richtlinie erteilt wurden.

Angeben mehrerer Services im visuellen Editor

Wenn Sie mit dem visuellen Editor eine Richtlinie erstellen, können Sie nur jeweils einen Service auswählen. Diese Methode hat sich bewährt, da der visuelle Editor Ihnen dann die Auswahl von Aktionen für diesen Service gestattet. Sie wählen dann aus den Ressourcen aus, die vom

Service und den ausgewählten Aktionen unterstützt werden. Dies vereinfacht das Erstellen und die Problembehebung Ihrer Richtlinie.

Wenn Sie mit der JSON-Syntax vertraut sind, können Sie mit einem Platzhalter (*) mehrere Services manuell angeben. Geben Sie beispielsweise **`Code*`** ein, um Berechtigungen für alle Services bereitzustellen, die mit `Code` beginnen, beispielsweise `CodeBuild` und `CodeCommit`. Allerdings müssen Sie dann die Aktionen und Ressourcen-ARNs eingeben, um die Richtlinie zu vervollständigen. Wenn Sie darüber hinaus Ihre Richtlinien speichern, wird diese möglicherweise umstrukturiert, um jeden Service in einen separaten Berechtigungsblock einzuschließen.

Wenn Sie die JSON-Syntax (beispielsweise Platzhalter) für Services verwenden möchten, erstellen, bearbeiten und speichern Sie Ihre Richtlinie über die Editoroption JSON.

Reduzieren der Größe Ihrer Richtlinie im visuellen Editor

Wenn Sie mit dem visuellen Editor eine Richtlinie erstellen, erstellt IAM ein JSON-Dokument zum Speichern der Richtlinie. Sie können dieses Dokument anzeigen, indem Sie zur Editoroption JSON wechseln. Wenn das JSON-Dokument die zulässige Größe für eine Richtlinie überschreitet, zeigt der visuelle Editor eine Fehlermeldung an und Sie können Ihrer Richtlinie weder anzeigen noch speichern. Informationen zur IAM-Größenbeschränkung für verwaltete Richtlinien finden Sie unter IAM- und STS-Zeichenlimits.

Zur Reduzierung der Größe Ihrer Richtlinie im visuellen Editor bearbeiten Sie die Richtlinie oder verschieben Sie Berechtigungsblöcke in eine andere Richtlinie. Die Fehlermeldung enthält die Anzahl der Zeichen in Ihrem Richtliniendokument. Anhand dieser Informationen können Sie die Größe reduzieren.

Beheben von Problemen aufgrund nicht erkannter Services, Aktionen oder Ressourcentypen im visuellen Editor

Wenn Sie eine Richtlinie im visuellen Editor erstellen oder bearbeiten, wird möglicherweise eine Warnung angezeigt, aus der hervorgeht, dass Ihre Richtlinie einen nicht erkannten Service, eine nicht erkannte Aktion oder einen nicht erkannten Ressourcentyp enthält.

> **Note**
>
> IAM überprüft Servicenamen, Aktionen und Ressourcentypen für Services, die Richtlinienübersichten unterstützen. Ihre Richtlinienübersicht kann jedoch einen nicht

vorhandenen Ressourcenwert oder eine nicht vorhandene Bedingung enthalten. Testen Sie die Richtlinien grundsätzlich mit dem Richtliniensimulator.

Wenn Ihre Richtlinie nicht erkannte Services, Aktionen oder Ressourcentypen enthält, ist einer der folgenden Fehler aufgetreten:

- Vorversion-Service – Services, die sich in der Vorversion befinden, unterstützen den visuellen Editor nicht. Wenn Sie mit der Vorversion arbeiten, können Sie die Warnung ignorieren und fortfahren. Allerdings müssen Sie die Aktionen und Ressourcen-ARNs manuell eingeben, um Ihre Richtlinie zu vervollständigen. Alternativ können Sie die Editoroption JSON auswählen oder ein JSON-Richtliniendokument eingeben oder einfügen.
- Benutzerdefinierter Service – Benutzerdefinierte Services unterstützen den visuellen Editor nicht. Wenn Sie einen benutzerdefinierten Service nutzen, können Sie die Warnung ignorieren und fortfahren. Allerdings müssen Sie die Aktionen und Ressourcen-ARNs manuell eingeben, um Ihre Richtlinie zu vervollständigen. Alternativ können Sie die Editoroption JSON auswählen oder ein JSON-Richtliniendokument eingeben oder einfügen.
- Service unterstützt den visuellen Editor nicht – Wenn Ihre Richtlinie einen allgemein verfügbaren (GA) Service enthält, der den visuellen Editor nicht unterstützt, können Sie die Warnung ignorieren und fortfahren, müssen aber die Aktionen und Ressourcen-ARNs manuell eingeben, um Ihre Richtlinie zu vervollständigen. Alternativ können Sie die Editoroption JSON auswählen oder ein JSON-Richtliniendokument eingeben oder einfügen.

  Normalerweise handelt es sich bei verfügbaren Services um veröffentlichte Services und nicht um Vorversion- oder benutzerdefinierte Services. Wenn ein nicht erkannter Service allgemein verfügbar und der Name richtig geschrieben ist, unterstützt der Service den visuellen Editor nicht. Weitere Informationen zur Inanspruchnahme des Supports für den visuellen Editor oder der Richtlinienübersicht in Bezug auf einen allgemein verfügbaren GA-Service finden Sie unter Der Service unterstützt keine IAM-Richtlinienübersichten.
- Aktion unterstützt den visuellen Editor nicht – Wenn Ihre Richtlinie einen unterstützten Service mit einer nicht unterstützten Aktion enthält, können Sie die Warnung ignorieren und fortfahren, müssen aber die Ressourcen-ARNs manuell eingeben, um Ihre Richtlinie zu vervollständigen. Alternativ können Sie die Editoroption JSON auswählen oder ein JSON-Richtliniendokument eingeben oder einfügen.

  Wenn Ihre Richtlinie einen unterstützten Service mit einer nicht unterstützten Aktion enthält, unterstützt der Service den visuellen Editor nicht vollständig. Weitere Informationen zur

Inanspruchnahme des Supports für den visuellen Editor oder der Richtlinienübersicht in Bezug auf einen allgemein verfügbaren GA-Service finden Sie unter Der Service unterstützt keine IAM-Richtlinienübersichten.

- Ressourcen-Typ unterstützt den visuellen Editor nicht – Wenn Ihre Richtlinie eine unterstützte Aktion mit einem nicht unterstützten Ressourcentyp enthält, können Sie die Warnung ignorieren und fortfahren. Allerdings kann IAM nicht bestätigen, dass Sie Ressourcen für all Ihre ausgewählten Aktionen eingeschlossen haben. Möglicherweise werden weitere Warnungen angezeigt.
- Tippfehler – Wenn Sie manuell einen Service, eine Aktion oder eine Ressource im visuellen Editor eingeben, erstellen Sie möglicherweise eine Richtlinie, die einen Tippfehler enthält. Um dies zu vermeiden, empfehlen wir Ihnen, den visuellen Editor zu verwenden, indem Sie aus der Liste der Services und Aktionen auswählen und dann den Ressourcenabschnitt entsprechend den Aufforderungen ausfüllen. Wenn ein Service den visuellen Editor allerdings nicht vollständig unterstützt, müssen Sie ggf. Teile der Richtlinie manuell eingeben.

  Wenn Sie sicher sind, dass Ihre Richtlinie keinen der oben genannten Fehler enthält, handelt es sich möglicherweise um einen Tippfehler. Überprüfen Sie, ob Service-, Aktions- oder Ressourcentypen-Namen falsch geschrieben wurden. Beispielsweise können Sie `s2` anstelle von `s3` und `ListMyBuckets` anstelle von `ListAllMyBuckets` verwenden. Ein weiterer gewöhnlicher Aktionstippfehler ist die Einbeziehung von unnötigem Text in ARNs, wie z. B. `arn:aws:s3: : :*`, oder fehlende Doppelpunkte in Aktionen, wie z. B. `iam.CreateUser`. Sie können eine Richtlinie evaluieren, die möglicherweise Tippfehler enthält, indem Sie Next (Weiter) auswählen, um die Richtlinienübersicht anzuzeigen und sicherzustellen, dass die Richtlinie die von Ihnen beabsichtigten Berechtigungen erteilt.

Problembehandlung unter Verwendung von Richtlinienübersichten

Sie können Probleme diagnostizieren und beheben, die im Zusammenhang mit der Richtlinienübersicht auftreten.

Übersicht fehlender Richtlinien

Die IAM-Konsole enthält Tabellen mit Richtlinienübersichten, die die Zugriffsebene, Ressourcen und Bedingungen aufführt, die für die einzelnen Services in einer Richtlinie zugelassen oder verweigert werden. Richtlinien werden in drei Tabellen zusammengefasst: Richtlinienübersicht, Serviceübersicht und Aktionsübersicht. In der Tabelle Richtlinienübersicht werden die Services und die Übersichten der für die ausgewählte Richtlinie definierten Berechtigungen aufgelistet.

Sie können die Zusammenfassung der Richtlinien für alle Richtlinien, die einer Entität zugeordnet sind, auf der Seite Policy details (Richtliniendetails) für diese Richtlinie einsehen. Sie können die Richtlinienübersicht für alle verwalteten Richtlinien auf der Seite Policies (Richtlinien) einsehen. Wenn AWS keine Zusammenfassung für eine Richtlinie gerendert werden kann, wird anstelle der Zusammenfassung das JSON-Richtliniendokument angezeigt, und es wird die folgende Fehlermeldung angezeigt:

A summary for this policy cannot be generated. (Eine Zusammenfassung für diese Richtlinie kann nicht erstellt werden.) You can still view or edit the JSON policy document. (Sie können das JSON-Richtliniendokument weiterhin anzeigen oder bearbeiten.)

Wenn Ihre Richtlinie keine Übersicht enthält, ist einer der folgenden Fehler aufgetreten:

- Nicht unterstütztes Richtlinienelement – IAM unterstützt nicht die Erstellung von Richtlinienübersichten, die eines der folgenden Richtlinienelemente enthalten:
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Keine Richtlinienberechtigung – Wenn eine Richtlinie keine effektiven Berechtigungen erteilt, kann keine Richtlinienübersicht erstellt werden. Wenn eine Richtlinie beispielsweise ein einzelnes Statement mit dem Element `"NotAction": "*"` enthält, wird Zugriff auf alle Aktionen mit Ausnahme von "allen Aktionen" (*) gewährt. Das bedeutet, dass `Deny`- oder `Allow`-Zugriff auf nichts gewährt wird.

  > **Note**
  >
  > Sie müssen bei der Verwendung dieser Richtlinienelemente wie `NotPrincipal`, `NotAction` und `NotResource` Vorsicht walten lassen. Weitere Informationen zur Verwendung von Richtlinienelementen finden Sie unter IAM-JSON-Richtlinienelementreferenz.

  Wenn Sie nicht übereinstimmende Services und Ressourcen angeben, erstellen Sie eine Richtlinie, die keine effektiven Berechtigungen gewährt. Dies kann der Fall sein, wenn Sie in einem Service Aktionen und Ressourcen aus einem anderen Service definieren. In diesem Fall erscheint die Richtlinienübersicht. Der einzige Hinweis darauf, dass ein Problem vorliegt, ist der Umstand, dass in der Ressourcen-Spalte eine Ressource eines anderen Services enthalten ist. Wenn diese Spalte

eine nicht übereinstimmende Ressource enthält, sollten Sie Ihre Richtlinien auf Fehler überprüfen. Um Ihre Richtlinien besser zu verstehen, testen Sie sie stets mit dem Richtliniensimulator.

Die Richtlinienübersicht umfasst nicht erkannte Services, Aktionen oder Ressourcentypen.

In der IAM-Konsole, wenn in der Richtlinienübersicht ein Warnsymbol
()
erscheint, kann die Richtlinie einen Service, eine Aktion oder einen Ressourcentyp enthalten, der bzw. die nicht erkannt wird. Weitere Informationen zu Warnungen innerhalb einer Richtlinienübersicht finden Sie unter Richtlinienübersicht (Liste der Services).

> **Note**
>
> IAM überprüft Servicenamen, Aktionen und Ressourcentypen für Services, die Richtlinienübersichten unterstützen. Ihre Richtlinienübersicht kann jedoch einen nicht vorhandenen Ressourcenwert oder eine nicht vorhandene Bedingung enthalten. Testen Sie die Richtlinien grundsätzlich mit dem Richtliniensimulator.

Wenn Ihre Richtlinie nicht erkannte Services, Aktionen oder Ressourcentypen enthält, ist einer der folgenden Fehler aufgetreten:

- Vorversion-Service – Services, die sich in der Vorversion befinden und keine Richtlinienübersicht unterstützen.
- Benutzerdefinierter Service – Richtlinienübersichten werden von benutzerdefinierten Services nicht unterstützt.
- Service unterstützt keine Übersichten – Wenn Ihre Richtlinie einen allgemein verfügbaren (generally available – GA) Service umfasst, der keine Richtlinienübersichten unterstützt, wird der Service in den Abschnitt Unrecognized services (Nicht erkannte Services) der Richtlinienübersichtstabelle aufgenommen. Normalerweise handelt es sich bei verfügbaren Services um veröffentlichte Services und nicht um Vorversion- oder benutzerdefinierte Services. Wenn ein nicht erkannter Service allgemein verfügbar und der Name richtig geschrieben ist, unterstützt der Service keine IAM-Richtlinienübersichten. Weitere Informationen zur Unterstützungsanforderung von Richtlinienübersichten für allgemein verfügbare GA-Services finden Sie unter Der Service unterstützt keine IAM-Richtlinienübersichten.

- Aktion unterstützt keine Übersichten – Wenn Ihre Richtlinie einen unterstützten Service mit einer nicht unterstützten Aktion enthält, wird die Aktion in den Abschnitt Unrecognized actions (Nicht erkannte Services) der Service-Übersichtstabelle aufgenommen. Weitere Informationen zu Warnungen innerhalb einer Serviceübersicht finden Sie unter Serviceübersicht (Liste der Aktionen).
- Ressourcentyp unterstützt keine Übersichten – Wenn Ihre Richtlinie eine unterstützte Aktion mit einem nicht unterstützten Ressourcentyp enthält, wird die Ressource in den Abschnitt Unrecognized resource types (Nicht erkannte Servicetypen) der Service-Übersichtstabelle aufgenommen. Weitere Informationen zu Warnungen innerhalb einer Serviceübersicht finden Sie unter Serviceübersicht (Liste der Aktionen).
- Tippfehler — AWS überprüft, ob das JSON syntaktisch korrekt ist und ob die Richtlinie keine Tippfehler oder andere Fehler im Rahmen der Richtlinienvalidierung enthält.

> **Note**
>
> Als bewährte Methode empfehlen wir, IAM Access Analyzer zu verwenden, um Ihre IAM-Richtlinien zu validieren und sichere und funktionale Berechtigungen zu gewährleisten. Wir empfehlen, vorhandene Richtlinien zu öffnen und alle Richtlinienvalidierungen zu überprüfen und zu lösen.

Der Service unterstützt keine IAM-Richtlinienübersichten

Wenn ein allgemein verfügbarer Service (GA) oder eine Aktionen nicht von den IAM-Richtlinienübersichten oder dem visuellen Editor erkannt werden, werden diese Funktionen möglicherweise vom Service nicht unterstützt. Allgemein verfügbare Services sind veröffentlichte Services, keine Vorversion- oder benutzerdefinierte Services. Wenn ein nicht erkannter Service allgemein verfügbar und der Name richtig geschrieben ist, unterstützt der Service diese Funktionen nicht. Wenn Ihre Richtlinie einen unterstützten Service mit einer nicht unterstützten Aktion enthält, werden die IAM-Richtlinienübersichten nicht vollständig von diesem Service unterstützt.

So können Sie anfordern, dass von einem Service die Unterstützung der IAM-Richtlinienübersicht oder die des visuellen Editors hinzugefügt wird

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Suchen Sie die Richtlinie, die den nicht unterstützten Service enthält:

- Wenn es sich um eine verwaltete Richtlinie handelt, wählen Sie im Navigationsbereich Policies (Richtlinien). Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie anzeigen möchten.
- Wenn es sich um eine Inlinerichtlinie handelt, die dem Benutzer zugeordnet ist, wählen Sie im Navigationsbereich Users (Benutzer). Wählen Sie in der Benutzerliste den Namen des Benutzers aus, dessen Richtlinie Sie anzeigen möchten. Erweitern Sie in der Richtlinientabelle des Benutzers den Header für die anzuzeigende Richtlinienübersicht.

3. Wählen Sie links in der AWS Management Console Fußzeile Feedback aus. Geben Sie im Feld Feedback for IAM (Feedback für IAM) **`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`** ein. Wenn Sie möchten, dass Übersichten von mehreren Services unterstützt werden sollen, geben Sie ein **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

So können Sie die Unterstützung der IAM-Richtlinienübersicht für eine fehlende Aktion durch einen Service hinzufügen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Suchen Sie die Richtlinie, die den nicht unterstützten Service enthält:

   - Wenn es sich um eine verwaltete Richtlinie handelt, wählen Sie im Navigationsbereich Policies (Richtlinien). Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie anzeigen möchten.
   - Wenn es sich um eine Inlinerichtlinie handelt, die dem Benutzer zugeordnet ist, wählen Sie im Navigationsbereich Users (Benutzer). Wählen Sie in der Benutzerliste den Namen des Benutzers aus, dessen Richtlinie Sie anzeigen möchten. Wählen Sie in der Richtlinientabelle des Benutzers den Namen der anzuzeigenden Richtlinie aus, um die Richtlinienübersicht zu erweitern.

3. Wählen Sie in der Richtlinienübersicht den Namen des Services aus, der eine nicht unterstützte Aktion enthält.
4. Wählen Sie links in der AWS Management Console Fußzeile Feedback aus. Geben Sie im Feld Feedback for IAM (Feedback für IAM) **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`** ein. Wenn Sie mehr als eine nicht unterstützte Aktion melden

möchten, geben Sie Folgendes ein **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**.

Um fehlende Aktionen eines anderen Services hinzuzufügen, wiederholen Sie die letzten drei Schritte.

Meine Richtlinie erteilt nicht die erwarteten Berechtigungen

Um einem Benutzer, einer Gruppe, Rolle oder Ressource Berechtigungen zuzuordnen, müssen Sie eine Richtlinie erstellen. Dabei handelt es sich um ein Dokument, das Berechtigungen definiert. Das Richtliniendokument umfasst die folgenden Elemente:

- Effect – gibt an, ob die Richtlinie den Zugriff erlaubt oder verweigert
- Action – die Liste der Aktionen, die durch die Richtlinie erlaubt oder verweigert werden
- Resource – die Liste der Ressourcen, für die die Aktionen durchgeführt werden können
- Condition (Optional) – die Bedingungen, unter denen die Richtlinie die Berechtigung erteilt

Weitere Informationen zu diesen und anderen Richtlinienelementen finden Sie unter IAM-JSON-Richtlinienelementreferenz.

Wenn Sie den Zugriff gewähren, muss Ihre Richtlinie eine Aktion mit einer unterstützten Ressource definieren. Wenn Ihre Richtlinie außerdem eine Bedingung umfasst, muss die Bedingung einen globalen Bedingungsschlüssel enthalten oder sich auf die Aktion beziehen. Informationen darüber, welche Ressourcen von einer Aktion unterstützt werden, finden Sie in der AWS -Dokumentation für Ihren Service. Informationen zu den Bedingungen, die von einer Aktion unterstützt werden, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Um zu erfahren, ob Ihre Richtlinie eine Aktion, Ressource oder Bedingung definiert, die keine Berechtigungen gewährt, können Sie die Richtlinienzusammenfassung für Ihre Richtlinie über die IAM-Konsole unter https://console.aws.amazon.com/iam/ anzeigen. Sie können mit Richtlinienzusammenfassungen Probleme in Ihrer Richtlinie identifizieren und beheben.

Es gibt mehrere Gründe, warum ein Element möglicherweise keine Berechtigungen erteilt, obwohl es in der IAM-Richtlinie definiert wird:

- Eine Aktion wurde ohne passende Ressource definiert.
- Eine Ressource wurde ohne passende Aktion definiert.

- Eine Bedingung wurde ohne passende Aktion definiert.

Beispiele für Richtlinienzusammenfassungen einschließlich Warnungen finden Sie unter the section called “Richtlinienübersicht (Liste der Services)”.

Eine Aktion wurde ohne passende Ressource definiert

Die nachstehende Richtlinie definiert alle `ec2:Describe*`-Aktionen sowie eine spezifische Ressource. Keine der `ec2:Describe`-Aktionen wird gewährt, da keine dieser Aktionen Berechtigungen auf Ressourcenebene unterstützt. Berechtigungen auf Ressourcenebene zeigen an, dass die Aktion Ressourcen mit ARNs im `Resource`-Element der Richtlinie unterstützt. Wenn eine Aktion keine Berechtigungen auf Ressourcenebene unterstützt, muss diese Anweisung in der Richtlinie einen Platzhalter (`*`) im `Resource`-Element verwenden. Informationen darüber, welche Services Berechtigungen auf Ressourcenebene unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID:instance/*"
    }]
}
```

Diese Richtlinie stellt keine Berechtigungen bereit und die Richtlinienzusammenfassung enthält den folgenden Fehler:

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

Um diese Richtlinie zu reparieren, müssen Sie `*` im `Resource`-Element verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
```

```
}
```

Eine Ressource wurde ohne passende Aktion definiert

Die nachstehende Richtlinie definiert einen Amazon S3-Bucket, enthält jedoch keine S3-Aktion, die auf diese Ressource angewendet werden kann. Diese Richtlinie gewährt auch vollen Zugriff auf alle CloudFront Amazon-Aktionen.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Diese Richtlinie bietet Berechtigungen für alle CloudFront Aktionen. Da die Richtlinie jedoch die S3-Ressource `examplebucket` definiert, ohne S3-Aktionen zu definieren, enthält die Richtlinienzusammenfassung die folgende Warnung:

`This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Um diese Richtlinie zu reparieren, damit sie S3-Bucket-Berechtigungen bereitstellt, müssen Sie S3-Aktionen definieren, die auf eine Bucket-Ressource angewendet werden können.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "cloudfront:*",
            "s3:CreateBucket",
            "s3:ListBucket*",
            "s3:PutBucket*",
            "s3:GetBucket*"
```

```
        ],
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Um diese Richtlinie dahingehend zu korrigieren, dass nur CloudFront Berechtigungen bereitgestellt werden, entfernen Sie alternativ die S3-Ressource.

Eine Bedingung wurde ohne passende Aktion definiert

In der nachstehenden Richtlinie werden zwei Amazon S3-Aktionen für alle S3-Ressourcen definiert, wenn das S3-Präfix `custom` und die Versions-ID `1234` lautet. Der Bedingungsschlüssel `s3:VersionId` wird jedoch zum Markieren der Objektversion verwendet und wird von den definierten Bucket-Aktionen nicht unterstützt. Informationen darüber, welche Bedingungen von einer Aktion unterstützt werden, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Folgen Sie dem Link zur Servicedokumentation für Bedingungsschlüssel.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
```

```
}
```

Diese Richtlinie stellt Berechtigungen für die Aktionen `s3:ListBucketVersions` und `s3:ListBucket` bereit, wenn der Bucket-Name das Präfix `custom` enthält. Da die Bedingung `s3:VersionId` jedoch von keiner der definierten Aktionen unterstützt wird, enthält die Richtlinienzusammenfassung den folgenden Fehler:

```
This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.
```

Um diese Richtlinie so zu reparieren, dass sie die Markierung der S3-Objektversion verwendet, müssen Sie eine S3-Aktion definieren, die den Bedingungsschlüssel `s3:VersionId` unterstützt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Diese Richtlinie stellt Berechtigungen für jede Aktion und Bedingung in der Richtlinie bereit. Die Richtlinie stellt jedoch noch keine Berechtigungen bereit, da es nicht vorkommt, dass eine einzige Aktion beide Bedingungen erfüllt. Stattdessen müssen Sie zwei separate Anweisungen erstellen, die jeweils nur Aktionen mit den Bedingungen enthält, auf die sie zutreffen.

Um diese Richtlinie zu reparieren, erstellen Sie zwei Anweisungen. Die erste Anweisung enthält die Aktionen, die die Bedingung `s3:prefix` unterstützen, und die zweite Anweisung enthält die Aktionen, die die Bedingung `s3:VersionId` unterstützen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Fehlerbehebung bei der Richtlinienverwaltung

Sie können Probleme diagnostizieren und beheben, die im Zusammenhang mit der Richtlinienverwaltung auftreten.

Anfügen oder Entfernen einer Richtlinie zu/von einem IAM-Konto

Einige AWS verwaltete Richtlinien sind mit einem Dienst verknüpft. Diese Richtlinien werden nur mit einer serviceverknüpften Rolle für diesen Service verwendet. In der IAM-Konsole ist auf der

Seite Policy details (Richtliniendetails) für eine Richtlinie ein Banner enthalten, um anzuzeigen, dass die Richtlinie mit einem Service verknüpft ist. Sie können diese Richtlinie zu einem Benutzer, einer Gruppe oder einer Rolle in IAM anfügen. Beim Erstellen einer serviceverknüpften Rolle wird diese Richtlinie automatisch zu der neuen Rolle angefügt. Da die Richtlinie erforderlich ist, können Sie sie nicht von der serviceverknüpften Rolle trennen.

Ändern von Richtlinien für Ihre IAM-Identitäten basierend auf ihrer Aktivität

Sie können Richtlinien für Ihre IAM-Identitäten (Benutzer, Gruppen und Rollen) basierend auf ihren Aktivitäten aktualisieren. Sehen Sie sich dazu die Ereignisse Ihres Kontos im CloudTrail Eventverlauf an. CloudTrail Die Ereignisprotokolle enthalten detaillierte Ereignisinformationen, anhand derer Sie die Berechtigungen der Richtlinie ändern können. Möglicherweise stellen Sie fest, dass ein Benutzer oder eine Rolle versucht, eine Aktion in auszuführen, AWS und diese Anfrage abgelehnt wird. In diesem Fall können Sie überlegen, dem Benutzer oder der Rolle die Berechtigung zum Ausführen dieser Aktion zu gewähren. Wenn dies der Fall ist, können Sie ihrer Richtlinie die Aktion und sogar den ARN der Ressource hinzufügen, auf die sie zuzugreifen versuchen. Wenn der Benutzer oder die Rolle über Berechtigungen verfügt, die er bzw. sie nicht verwendet, können Sie alternativ in Betracht ziehen, diese Berechtigungen aus ihrer Richtlinie zu entfernen. Stellen Sie sicher, dass Ihre Richtlinien die geringsten Rechte gewähren, die erforderlich sind, um nur die notwendigen Aktionen durchzuführen. Weitere Informationen zur Verwendung CloudTrail finden Sie im AWS CloudTrail Benutzerhandbuch unter CloudTrail Ereignisse in der CloudTrail Konsole anzeigen.

Fehlerbehebung bei JSON-Richtliniendokumenten

Sie können Probleme diagnostizieren und beheben, die im Zusammenhang mit JSON-Richtliniendokumenten auftreten.

Validieren Ihrer Richtlinien

Wenn Sie eine JSON-Richtlinie erstellen oder bearbeiten, kann IAM eine Richtlinienvalidierung durchführen, um Ihnen beim Erstellen einer effektiven Richtlinie zu helfen. IAM identifiziert JSON-Syntaxfehler, während IAM Access Analyzer zusätzliche Richtlinienüberprüfungen mit Empfehlungen zur weiteren Verfeinerung Ihrer Richtlinien bietet. Weitere Informationen zur Richtlinienvalidierung finden Sie unter Validieren von IAM-Richtlinien. Weitere Informationen zu den Richtlinienvalidierungen von IAM Access Analyzer-Richtlinien und Empfehlungen erhalten Sie unter Richtlinienvalidierung von IAM Access Analyzer.

Ich habe keine Berechtigungen für die Richtlinienvalidierung im JSON-Editor

In der erhalten Sie möglicherweise die folgende Fehlermeldung AWS Management Console, wenn Sie nicht berechtigt sind, die Ergebnisse der IAM Access Analyzer-Richtlinienvalidierung anzuzeigen:

```
You need permissions. You do not have the permissions required to perform this operation. Ask your administrator to add permissions.
```

Um diesen Fehler zu beheben, bitten Sie Ihren Administrator, diese `access-analyzer:ValidatePolicy`-Berechtigung für Sie hinzuzufügen.

Mehrere JSON-Richtlinienobjekte

Eine IAM-Richtlinie darf nur aus einem JSON-Objekt bestehen. Ein Objekt wird mithilfe von {}-Klammern definiert. Obwohl Sie andere Objekte innerhalb eines JSON-Objekts verschachteln können, indem Sie zusätzliche { }-Klammern in das äußere Paar einbetten, kann eine Richtlinie nur ein äußerstes Paar von { }-Klammern enthalten. Das folgende Beispiel ist falsch. Es enthält zwei Objekte auf oberster Ebene (*Rot* hervorgehoben):

```
{
      "Version": "2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
     "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
     }
   }
```

Mit der richtigen Schreibweise können Sie das vorgenannte Beispiel jedoch in eine korrekte Richtlinie umwandeln. Statt zwei vollständige Richtlinienobjekte mit jeweils eigenen `Statement`-Elementen zu nutzen, können Sie die beiden Blöcke in einem einzelnen `Statement`-Element zusammenfassen. Das `Statement`-Element weist ein Array mit zwei Objekten als Wert auf, wie im folgenden Beispiel gezeigt (in Fettdruck hervorgehoben):

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":" *"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::my-bucket/*"
        }
      ]
    }
```

Mehrere JSON-Anweisungselemente

Dieser Fehler kann zunächst als eine Variante des Fehlers aus dem vorherigen Abschnitt interpretiert werden. Syntaktisch handelt es sich jedoch um einen anderen Fehler. Im folgenden Beispiel befindet sich nur ein Richtlinienobjekt auf der obersten Ebene (durch die {}-Klammern definiert). Das Objekt enthält jedoch zwei `Statement`-Elemente.

Eine IAM-Richtlinie darf nur ein `Statement`-Element enthalten. Dies setzt sich aus dem Namen (`Statement`) links vom Doppelpunkt und dem Wert rechts vom Doppelpunkt zusammen. Der Wert eines `Statement`-Elements muss ein Objekt sein (durch {}-Klammern definiert). Es muss ein `Effect`-Element, ein `Action`-Element und ein `Resource`-Element enthalten. Das folgende Beispiel ist falsch, da das Richtlinienobjekt zwei `Statement`-Elemente enthält (in *Rot* hervorgehoben):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
```

```
    }
```

Ein Wert-Objekt kann ein Array mehrerer Wert-Objekte sein. Um dieses Problem zu lösen, fassen Sie die zwei `Statement`-Elemente in einem Element mit einem Objekt-Array zusammen, wie im folgenden Beispiel gezeigt (in Fettdruck hervorgehoben):

```
{
      "Version": "2012-10-17",
      "Statement": [    {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::my-bucket/*"
        }
     ]
    }
```

Der Wert des `Statement`-Elements ist eine Objekt-Gruppe. Die Gruppe in diesem Beispiel besteht aus zwei Objekten. Jedes Objekt ist ein gültiger Wert für ein `Statement`-Element. Die Objekte in der Gruppe werden durch Kommas getrennt.

Mehrere Effect-, Action- oder Resource-Elemente in einem JSON-Anweisungselement

Auf der Wert-Seite des `Statement`-Namen/Wert-Paares darf das Objekt nur ein `Effect`-Element, ein `Action`-Element und ein `Resource`-Element enthalten. Das folgende Beispiel ist falsch, weil das Wert-Objekt des `Effect` zwei `Statement`-Elemente enthält:

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

> **Note**
>
> Die Richtlinien-Engine lässt solche Fehler in neuen oder bearbeiteten Richtlinien nicht zu. Die Richtlinien-Engine lässt jedoch weiterhin Richtlinien zu, die vor der Aktualisierung der Engine gespeichert wurden. Das Fehlerverhalten von vorhandenen Richtlinien ist wie folgt:
>
> - Mehrere `Effect`-Elemente: Nur das letzte `Effect`-Element wird berücksichtigt. Alle anderen werden ignoriert.
> - Mehrere `Action`-Elemente: Alle `Action`-Elemente werden intern kombiniert und als eine einzige Liste behandelt.
> - Mehrere `Resource`-Elemente: Alle `Resource`-Elemente werden intern kombiniert und als eine einzige Liste behandelt.
>
> Die Richtlinien-Engine gestattet es nicht, Richtlinien mit Syntax-Fehlern zu speichern. Sie müssen die Fehler in der Richtlinie korrigieren, um sie speichern zu können. Wir empfehlen, alle Empfehlungen für Richtlinienvalidierung für Ihre Richtlinien zu korrigieren.

Grundsätzlich besteht die Lösung darin, das falsche, überflüssige Element zu entfernen. Für `Effect`-Elemente ist dies ganz einfach: Wenn Sie im vorherigen Beispiel die Berechtigungen für Amazon EC2-Instances verweigern möchten, müssen Sie die Zeile `"Effect": "Allow",` aus der Richtlinie entfernen, wie nachfolgend dargestellt:

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

Wenn es sich jedoch bei dem duplizierten Element um `Action` oder `Resource` handelt, kann sich die Lösung schwieriger gestalten. Sie verfügen ggf. über mehrere Aktionen, die Sie zulassen (oder verweigern) möchten, oder Sie möchten ggf. die Berechtigung für den Zugriff auf mehrere Ressourcen kontrollieren. Das folgende Beispiel ist falsch, da es mehrere `Resource`-Elemente enthält (in *`Rot`* hervorgehoben):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
    }
```

Jedes der erforderlichen Elemente in einem Wert-Paar-Objekt eines `Statement`-Elements darf nur einmal vorhanden sein. Die Lösung besteht darin, jeden Wert in einem Array anzuordnen. Das folgende Beispiel erläutert dies, indem die beiden Ressourcenelemente in ein `Resource`-Element überführt werden, das ein separates Array als Wertobjekt aufweist (in Fettdruck hervorgehoben):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::my-bucket",
          "arn:aws:s3:::my-bucket/*"
        ]
      }
    }
```

Fehlendes JSON-Versionselement

Das Richtlinienelement `Version` unterscheidet sich von einer Richtlinienversion. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Andererseits wird eine Richtlinienversion erstellt, wenn Sie in IAM eine benutzerdefinierte, verwaltete Richtlinie bearbeiten. Die vorhandene Richtlinie wird von der geänderten Richtlinie nicht überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. Weitere Informationen zum Richtlinienelement `Version` finden Sie unter IAM-JSON-Richtlinienelemente: Version. Weitere Informationen zu den Richtlinienversionen finden Sie unter the section called “Versioning von IAM-Richtlinien”.

Im AWS Zuge der Weiterentwicklung der Funktionen werden den IAM-Richtlinien neue Funktionen hinzugefügt, um diese Funktionen zu unterstützen. Zuweilen geht eine Aktualisierung der

Richtliniensyntax mit der Vergabe einer neuen Versionsnummer einher. Wenn Sie in Ihren Richtlinien neuere Funktionen der Richtlinienformulierung verwenden, müssen Sie der Richtlinien-Parsing-Engine mitteilen, welche Version Sie verwenden. Die standardmäßige Richtlinienversion lautet "2008-10-17". Wenn Sie eine später hinzugefügte Richtlinienfunktion verwenden möchten, müssen Sie die Versionsnummer angeben, die die gewünschte Funktion unterstützt. Wir empfehlen, immer die neueste Versionsnummer der Richtliniensyntax zu verwenden, zurzeit `"Version": "2012-10-17"`. Die folgende Richtlinie ist beispielsweise falsch, da sie die Richtlinienvariable `${...}` im ARN für eine Ressource verwendet. Es fehlt aber die Angabe einer Richtliniensyntaxversion, die Richtlinienvariablen (*rot* hervorgehoben) unterstützt:

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Dieses Problem kann durch Hinzufügen eines `Version`-Elements am Anfang der Richtlinie mit dem Wert `2012-10-17`, die erste IAM-API-Version, die Richtlinienvariablen unterstützt, behoben werden (in Fettdruck hervorgehoben):

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Fehlerbehebung von FIDO-Sicherheitsschlüsseln

Verwenden Sie die hier aufgeführten Informationen, um häufige Probleme zu diagnostizieren und zu beheben, die beim Arbeiten mit FIDO2-Sicherheitsschlüsseln auftreten können.

Themen

Ich kann meinen FIDO-Sicherheitsschlüssel nicht aktivieren

Lesen Sie die folgenden Lösungen entsprechend Ihres Status als IAM-Benutzer oder -Systemadministrator

IAM-Benutzer

Wenn Sie Ihren FIDO-Sicherheitsschlüssel nicht aktivieren können, überprüfen Sie Folgendes:

- Verwenden Sie eine unterstützte Konfiguration?

  Informationen zu Geräten und Browsern, die Sie mit WebAuthn und verwenden können AWS, finden Sie unterUnterstützte Konfigurationen für die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln.
- Verwenden Sie Mozilla Firefox?

  Aktuelle Firefox-Versionen WebAuthn werden standardmäßig unterstützt. Gehen Sie wie folgt vor, um die Unterstützung für WebAuthn in Firefox zu aktivieren:

  1. Geben Sie in der Firefox-Adressleiste **`about:config`** ein.
  2. Geben Sie in der Suchleiste des sich öffnenden Bildschirms **`webauthn`** ein.
  3. Wählen Sie security.webauth.webauthn und ändern Sie den Wert auf true (wahr).
- Verwenden Sie Browser-Plugins?

  AWS unterstützt nicht die Verwendung von Plugins zum Hinzufügen von WebAuthn Browserunterstützung. Verwenden Sie stattdessen einen Browser, der den WebAuthn Standard nativ unterstützt.

  Auch wenn Sie einen unterstützten Browser verwenden, haben Sie möglicherweise ein Plugin, mit dem nicht kompatibel ist WebAuthn. Ein inkompatibles Plugin kann Sie daran hindern, Ihren FIDO-konformen Sicherheitsschlüssel zu aktivieren und zu verwenden. Sie sollten Plugins, die inkompatibel sein könnten, deaktivieren und Ihren Browser neu starten. Versuchen Sie dann erneut, den FIDO-Sicherheitsschlüssel zu aktivieren.

- Haben Sie die entsprechenden Berechtigungen?

  Wenn Sie keine der oben genannten Kompatibilitätsprobleme haben, verfügen Sie möglicherweise nicht über die entsprechenden Berechtigungen. Bitte wenden Sie sich an Ihren Systemadministrator.

Systemadministratoren

Wenn Sie ein Administrator sind und Ihre IAM-Benutzer ihre FIDO-Sicherheitsschlüssel trotz Verwendung einer unterstützten Konfiguration nicht aktivieren können, stellen Sie sicher, dass sie über die entsprechenden Berechtigungen verfügen. Ein ausführliches Beispiel finden Sie unter IAM-Tutorial: Zulassen, dass Ihre Benutzer ihre eigenen Anmeldeinformationen und MFA-Einstellungen konfigurieren können.

Ich kann mich mit meinem FIDO-Sicherheitsschlüssel nicht anmelden

Wenn Sie ein IAM-Benutzer sind und sich nicht AWS Management Console mit Ihrem FIDO-Sicherheitsschlüssel bei der anmelden können, finden Sie zunächst weitere Informationen unter. Unterstützte Konfigurationen für die Verwendung von Hauptschlüsseln und Sicherheitsschlüsseln Wenn Sie eine unterstützte Konfiguration verwenden, sich aber nicht anmelden können, wenden Sie sich an Ihren Systemadministrator, um Hilfe zu erhalten.

Ich habe meinen FIDO-Sicherheitsschlüssel verloren oder beschädigt

Einem Benutzer können bis zu acht MFA-Geräte einer beliebigen Kombination der derzeit unterstützten MFA-Typen zugewiesen werden. Bei mehreren MFA-Geräten ist nur ein MFA-Gerät erforderlich, um sich bei der AWS Management Console anzumelden. Das Ersetzen eines FIDO-Sicherheitsschlüssels ist vergleichbar mit dem Ersetzen eines Hardware-TOTP-Tokens. Informationen darüber, was zu tun ist, wenn Sie ein MFA-Gerät verlieren oder beschädigen, finden Sie unter Was passiert, wenn ein MFA-Gerät verloren geht oder nicht funktioniert?.

Sonstige Probleme

Wenn Sie ein Problem mit FIDO-Sicherheitsschlüsseln haben, das hier nicht behandelt wird, führen Sie einen der folgenden Schritte aus:

- IAM-Benutzer: Wenden Sie sich an Ihren Systemadministrator.
- AWS-Konto Stammbenutzer: Kontakt AWS Support.

Fehlerbehebung für IAM-Rollen

Verwenden Sie die hier aufgeführten Informationen, um häufige Probleme zu diagnostizieren und zu beheben, die beim Arbeiten mit IAM-Rollen auftreten können.

Themen

Ich kann eine Rolle nicht übernehmen

Überprüfen Sie, ob Folgendes der Fall ist:

- Damit Benutzer die aktuelle Rolle innerhalb einer Rollensitzung wieder übernehmen können, geben Sie den Rollen-ARN oder AWS-Konto ARN als Principal in der Rollenvertrauensrichtlinie an. AWS-Services die Rechenressourcen wie Amazon EC2, Amazon ECS, Amazon EKS und Lambda bereitstellen, stellen temporäre Anmeldeinformationen bereit und aktualisieren diese Anmeldeinformationen automatisch. Dadurch wird sichergestellt, dass Sie immer über gültige Anmeldeinformationen verfügen. Für diese Dienste ist es nicht erforderlich, die aktuelle Rolle erneut anzunehmen, um temporäre Anmeldeinformationen zu erhalten. Wenn Sie jedoch beabsichtigen, Sitzungs-Tags oder eine Sitzungsrichtlinie zu übergeben, müssen Sie die aktuelle Rolle erneut annehmen. Informationen zum Ändern einer Rollenvertrauensrichtlinie zum Hinzufügen der Hauptrollen ARN oder AWS-Konto ARN finden Sie unterÄndern einer Rollenvertrauensrichtlinie (Konsole).
- Wenn Sie mithilfe von eine Rolle übernehmen AWS Management Console, stellen Sie sicher, dass Sie den exakten Namen Ihrer Rolle verwenden. Bei Rollennamen wird die Groß-/Kleinschreibung beachtet.

- Wenn Sie mithilfe von AWS STS API oder eine Rolle annehmen AWS CLI, stellen Sie sicher, dass Sie den genauen Namen Ihrer Rolle im ARN verwenden. Bei Rollennamen wird die Groß-/Kleinschreibung beachtet.
- Überprüfen Sie, ob Ihre IAM-Richtlinie die Berechtigung zum Aufrufen der Aktion `sts:AssumeRole` für die Rolle gewährt, die Sie übernehmen möchten. Das `Action`-Element Ihrer IAM-Richtlinie muss es Ihnen ermöglichen, die Aktion `AssumeRole` aufzurufen. Darüber hinaus muss das `Resource`-Element Ihrer IAM-Richtlinie die Rolle angeben, die Sie übernehmen möchten. Das Element `Resource` kann beispielsweise eine Rolle anhand ihres Amazon-Ressourcennamens (ARN) oder durch einen Platzhalter (*) definieren. Sie müssen durch mindestens eine für Sie gültige Richtlinie Berechtigungen wie die folgenden erhalten:

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Vergewissern Sie sich, dass Ihre IAM-Identität mit allen Tags markiert ist, die die IAM-Richtlinie erfordert. In den folgenden Richtlinienberechtigungen erfordert das `Condition`-Element beispielsweise, dass Sie als Auftraggeber, der die Übernahme der Rolle beantragt, ein bestimmtes Tag haben müssen. Sie müssen mit dem Tag `department = HR` oder `department = CS` markiert sein. Andernfalls können Sie die Rolle nicht übernehmen. Weitere Informationen zum Markieren von IAM-Benutzern und Rollen finden Sie unter the section called “Markieren von IAM-Ressourcen”.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/department": [
            "HR",
            "CS"
        ]}}
```

- Stellen Sie sicher, dass Sie alle Bedingungen erfüllen, die in der Vertrauensrichtlinie der Rolle definiert sind. Eine `Condition` kann ein Ablaufdatum, eine externe ID oder eine bestimmte IP-Adresse festlegen, von der die Anforderung kommen muss. Sehen Sie sich das folgende Beispiel an: Wenn das aktuelle Datum nach dem angegebenen Datum liegt, sind die Bedingungen der Richtlinie nie erfüllt und die Berechtigung zum Übernehmen der Rolle kann nicht gewährt werden.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
```

```
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
    "Condition": {
        "DateLessThan" : {
            "aws:CurrentTime" : "2016-05-01T12:00:00Z"
        }
    }
```

- Stellen Sie sicher, dass es sich bei der Entität, AWS-Konto von der aus Sie anrufen, um eine vertrauenswürdige Entität für die Rolle `AssumeRole` handelt, die Sie übernehmen. Vertrauenswürdige Entitäten sind in der Vertrauensrichtlinie einer Rolle als `Principal` definiert. Das nachfolgende Beispiel enthält eine Vertrauensrichtlinie, die der Rolle angefügt ist, die Sie übernehmen möchten. In diesem Beispiel muss die Konto-ID des IAM-Benutzers, über den Sie sich angemeldet haben, 123456789012 lauten. Wenn Ihre Kontonummer nicht im `Principal`-Element der Vertrauensrichtlinie der Rolle aufgeführt ist, können Sie die Rolle nicht übernehmen. Es spielt keine Rolle, welche Berechtigungen Ihnen in den Zugriffsrichtlinien gewährt werden. Beachten Sie, dass in der Beispielrichtlinie nur Berechtigungen für Aktionen gewährt werden, die zwischen dem 01.07.2017 und dem 31.12.2017 (UTC) ausgeführt werden. Wenn Sie sich vor oder nach diesem Zeitraum anmelden, sind die Bedingungen der Richtlinie nicht erfüllt und Sie können die Rolle nicht übernehmen.

```
    "Effect": "Allow",
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
```

- Quellenidentität — Administratoren können Rollen so konfigurieren, dass Identitäten eine benutzerdefinierte Zeichenfolge übergeben müssen, die die Person oder Anwendung identifiziert, in der Aktionen ausgeführt werden. Diese wird als AWSQuellidentität bezeichnet. Überprüfen Sie, ob für die übernommene Rolle eine Quellidentität festgelegt ist. Weitere Informationen zu Quellidentität finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

In meinem AWS -Konto wird eine neue Rolle angezeigt

Für einige AWS Dienste ist es erforderlich, dass Sie eine bestimmte Art von Servicerolle verwenden, die direkt mit dem Dienst verknüpft ist. Diese serviceverknüpfte Rolle wird vom Service vordefiniert und enthält alle vom Service benötigten Berechtigungen. Dadurch wird das Einrichten eines Service

vereinfacht, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Allgemeine Informationen zu serviceverknüpften Rollen finden Sie unter Verwenden von serviceverknüpften Rollen.

Möglicherweise verwenden Sie einen Service bereits, wenn er damit beginnt, serviceverknüpfte Rollen zu unterstützen. In diesem Fall erhalten Sie möglicherweise eine E-Mail mit Hinweisen zu einer neuen Rolle in Ihrem Konto. Diese Rolle enthält alle Berechtigungen, die der Service benötigt, um in Ihrem Namen Aktionen auszuführen. Sie müssen keine Aktion durchführen, um diese Rolle zu unterstützen. Jedoch sollten Sie die Rolle nicht aus Ihrem Konto löschen. Andernfalls können eventuell Berechtigungen entfernt werden, die der Service für den Zugriff auf AWS -Ressourcen benötigt. Sie können die serviceverknüpften Rollen in Ihrem Konto anzeigen, indem Sie zur IAM-Seite Roles Seite der IAM-Konsole wechseln. Serviceverknüpfte Rollen werden mit dem Hinweis (Service-linked role) in der Spalte Trusted entities (Vertrauenswürdige Entitäten) der Tabelle angezeigt.

Informationen darüber, welche Services diese serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist. Weitere Informationen zur Verwendung von serviceverknüpften Rollen für einen Service erhalten Sie über den Link Yes.

Ich kann eine Rolle in meinem AWS-Konto nicht bearbeiten oder löschen

Sie können die Berechtigungen für einer serviceverknüpfte Rolle in IAM nicht löschen oder bearbeiten. Diese Rollen umfassen vordefinierte Vertrauensstellungen und Berechtigungen, die für den Service erforderlich sind, um Aktionen in Ihrem Namen durchführen zu können. Sie können die IAM-Konsole oder die API verwenden AWS CLI, um nur die Beschreibung einer dienstbezogenen Rolle zu bearbeiten. Sie können die serviceverknüpften Rollen in Ihrem Konto anzeigen, indem Sie zur IAM-Seite Roles der Konsole wechseln. Serviceverknüpfte Rollen werden mit dem Hinweis (Service-linked role) in der Spalte Trusted entities (Vertrauenswürdige Entitäten) der Tabelle angezeigt. Ein Banner auf der Seite Summary (Übersicht) für die Rolle zeigt ebenfalls an, dass es sich um eine serviceverknüpfte Rolle handelt. Sie können diese Rollen nur über den verknüpften Service verwalten und löschen, wenn dieser Service die Aktion unterstützt. Gehen Sie beim Ändern oder Löschen einer serviceverknüpften Rolle mit Bedacht vor, da Sie dadurch möglicherweise Berechtigungen entfernen, die der Service benötigt, um auf AWS -Ressourcen zuzugreifen.

Informationen darüber, welche Services diese serviceverknüpften Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes in der Spalte Service-Linked Role angegeben ist.

Ich bin nicht berechtigt, Folgendes auszuführen: iam: PassRole

Wenn Sie eine serviceverknüpfte Rolle erstellen, müssen Sie über die Berechtigung verfügen, diese Rolle an den Service weiterzuleiten. Einige Services erstellen automatisch eine serviceverknüpfte Rolle in Ihrem Konto, wenn Sie eine Aktion in diesem Service durchführen. Beispiel: Amazon EC2 Auto Scaling erstellt die serviceverknüpfte `AWSServiceRoleForAutoScaling`-Rolle für Sie, wenn Sie das erste Mal eine Auto Scaling-Gruppe erstellen. Wenn Sie versuchen, eine Auto Scaling-Gruppe ohne die `PassRole`-Berechtigung zu erstellen, erhalten Sie folgenden Fehler:

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Um diesen Fehler zu beheben, bitten Sie Ihren Administrator, diese `iam:PassRole`-Berechtigung für Sie hinzuzufügen.

Informationen dazu, welche Services serviceverknüpfte Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Um zu erfahren, ob ein Service automatisch eine serviceverknüpfte Rolle für Sie erstellt, wählen Sie den Link Ja aus, um die Dokumentation zur serviceverknüpften Rolle für den Service anzuzeigen.

Warum kann ich keine Rolle mit einer 12-Stunden-Sitzung übernehmen? (AWS CLI, AWS API)

Wenn Sie die AWS STS `AssumeRole*` API- oder `assume-role*` CLI-Operationen verwenden, um eine Rolle anzunehmen, können Sie einen Wert für den `DurationSeconds` Parameter angeben. Sie können einen Wert von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle angeben. Wenn Sie einen Wert höher als diese Einstellung angeben, schlägt die Operation fehl. Diese Einstellung kann einen Höchstwert von 12 Stunden haben. Wenn Sie beispielsweise eine Sitzungsdauer von zwölf Stunden angeben, Ihr Administrator aber die maximale Sitzungsdauer auf sechs Stunden festgelegt hat, schlägt die Operation fehl. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.

Wenn Sie Verketten verwenden (Verwendung einer Rolle, um eine zweite Rolle anzunehmen), ist Ihre Sitzung auf maximale eine Stunde begrenzt. Wenn Sie den `DurationSeconds`-Parameter verwenden, um einen Wert größer als eine Stunde anzugeben, schlägt die Operation fehl.

Ich erhalte einen Fehler, wenn ich versuche, Rollen in der IAM-Konsole zu wechseln

Die Informationen, die Sie auf der Seite Switch Role (Rolle wechseln) eingeben, müssen mit den Informationen für die Rolle übereinstimmen. Andernfalls schlägt der Vorgang fehl und Sie erhalten die folgende Fehlermeldung:

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Wenn dieser Fehler angezeigt wird, bestätigen Sie, dass die folgenden Informationen korrekt sind:

- Konto-ID oder Alias — Die AWS-Konto ID ist eine 12-stellige Zahl. Ihr Konto hat möglicherweise einen Alias, bei dem es sich um eine benutzerfreundliche Kennung wie Ihren Firmennamen handelt, der anstelle Ihrer AWS-Konto ID verwendet werden kann. Sie können entweder die Konto-ID oder den Alias in diesem Feld verwenden.
- Role name (Rollenname) – Bei Rollennamen wird die Groß-/Kleinschreibung beachtet. Die Konto-ID und der Rollenname müssen mit dem für die Rolle konfigurierten Wert übereinstimmen.

Wenn Sie weiterhin eine Fehlermeldung erhalten, wenden Sie sich an Ihren Administrator, um die vorherigen Informationen zu überprüfen. Möglicherweise schränkt die Rollenvertrauensrichtlinie oder die IAM-Benutzerrichtlinie Ihren Zugriff ein. Ihr Administrator kann die Berechtigungen für diese Richtlinien überprüfen.

Zu meiner Rolle gehört eine Richtlinie, die es mir erlaubt, eine Aktion durchzuführen, ich erhalte aber einen "Zugriff verweigert"-Fehler

Ihre Rollensitzung wird möglicherweise durch Sitzungsrichtlinien beschränkt. Wenn Sie temporäre Sicherheitsanmeldedaten programmgesteuert anfordern AWS STS, können Sie optional interne oder verwaltete Sitzungsrichtlinien übergeben. Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie programmgesteuert eine Sitzung mit temporären Anmeldeinformationen für eine Rolle erstellen. Sie können ein einzelnes JSON-Inline-Sitzungsrichtliniendokument mit dem `Policy`-Parameter übergeben. Sie können mit dem Parameter `PolicyArns` bis zu 10 verwaltete Sitzungsrichtlinien angeben. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Es kann auch sein, dass, wenn Ihr Administrator

oder ein benutzerdefiniertes Programm Ihnen temporäre Anmeldeinformationen bereitstellt, bereits eine Sitzungsrichtlinie zum Einschränken Ihres Zugriffs enthalten ist.

Der Service hat die Standardrichtlinienversion der Rolle nicht erstellt

Eine Servicerolle ist eine Rolle, die ein Service annimmt, um Aktionen in Ihrem Konto für Sie auszuführen. Wenn Sie einige AWS Serviceumgebungen einrichten, müssen Sie eine Rolle definieren, die der Dienst übernehmen soll. In einigen Fällen erstellt der Service die Servicerolle und die zugehörige Richtlinie in IAM für Sie. Obwohl Sie die Servicerolle und ihre Richtlinie innerhalb von IAM ändern oder löschen können, wird dies von AWS nicht empfohlen. Die Rolle und die Richtlinie sind nur für die Verwendung durch diesen Service bestimmt. Wenn Sie die Richtlinie bearbeiten und eine andere Umgebung einrichten, kann die Operation fehlschlagen, wenn der Service versucht, dieselbe Rolle und Richtlinie zu verwenden.

Wenn Sie den Dienst beispielsweise AWS CodeBuild zum ersten Mal verwenden, erstellt er eine Rolle mit dem Namen`codebuild-RWBCore-service-role`. Diese Servicerolle verwendet die Richtlinie namens `codebuild-RWBCore-managed-policy`. Wenn Sie die Richtlinie bearbeiten, wird eine neue Version erstellt und diese als Standardversion gespeichert. Wenn Sie einen nachfolgenden Vorgang in ausführen AWS CodeBuild, versucht der Dienst möglicherweise, die Richtlinie zu aktualisieren. Wenn dies der Fall ist, wird die folgende Fehlermeldung angezeigt:

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
RWBCore-service-role role. To continue, detach the policy from any other
identities and then delete the policy and the role.
```

Wenn dieser Fehler angezeigt wird, müssen Sie Änderungen in IAM vornehmen, bevor Sie mit der Serviceoperation fortfahren können. Legen Sie zunächst die Standardrichtlinienversion auf V1 fest, und versuchen Sie, die Operation erneut durchzuführen. Wenn V1 zuvor gelöscht wurde oder die Auswahl von V1 nicht funktioniert, bereinigen und löschen Sie die vorhandene Richtlinie und Rolle.

Weitere Informationen zum Bearbeiten verwalteter Richtlinien finden Sie unter Bearbeiten von kundenverwalteten Richtlinien (Konsole). Weitere Informationen zu Richtlinienversionen finden Sie unter Versioning von IAM-Richtlinien.

So löschen Sie eine Servicerolle und ihre 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 Policies.
3. Wählen Sie in der Richtlinienliste den Namen der Richtlinie aus, die sie löschen möchten.
4. Wählen Sie die Registerkarte Entities attached (Angefügte Entitäten) aus, um anzuzeigen, welche IAM-Benutzer, -Gruppen oder -Rollen diese Richtlinie verwenden. Wenn eine dieser Identitäten die Richtlinie verwendet, führen Sie die folgenden Aufgaben aus:

    a. Erstellen Sie eine neue verwaltete Richtlinie mit den erforderlichen Berechtigungen. Um sicherzustellen, dass die Identitäten vor und nach Ihren Aktionen dieselben Berechtigungen haben, kopieren Sie das JSON-Richtliniendokument aus der vorhandenen Richtlinie. Erstellen Sie dann die neue verwaltete Richtlinie, und fügen Sie das JSON-Dokument ein, wie unter Creating Policies using the JSON editor (Erstellen von Richtlinien mit dem JSON-Editor) beschrieben.

    b. Hängen Sie für jede betroffene Identität die neue Richtlinie an, und trennen Sie dann die alte davon. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen.
5. Wählen Sie im Navigationsbereich Roles.
6. Wählen Sie in der Liste der Rollen den Namen der Rolle aus, die Sie löschen möchten.
7. Wählen Sie die Registerkarte Trust relatiuonships (Vertrauensbeziehungen), um anzuzeigen, welche Entitäten die Rolle übernehmen können. Wenn eine andere Entität als der Service aufgeführt ist, führen Sie die folgenden Aufgaben aus:

    a. Erstellen Sie eine neue Rolle, die diesen Entitäten vertraut.

    b. Die Richtlinie, die Sie im vorherigen Schritt erstellt haben. Wenn Sie diesen Schritt übersprungen haben, erstellen Sie jetzt die neue verwaltete Richtlinie.

    c. Benachrichtigen Sie alle, die die Rolle angenommen haben darüber, dass dies jetzt nicht mehr möglich ist. Informieren Sie diese Personen darüber, wie Sie die neue Rolle übernehmen und über dieselben Berechtigungen verfügen können.
8. Löschen Sie die Richtlinie.
9. Löschen Sie die Rolle.

Es gibt keinen Anwendungsfall für eine Servicerolle in der Konsole

Bei einigen Diensten müssen Sie manuell eine Servicerolle erstellen, um dem Dienst Berechtigungen zum Durchführen von Aktionen in Ihrem Namen zu erteilen. Wenn der Dienst nicht in der IAM-Konsole aufgeführt ist, müssen Sie den Dienst manuell als vertrauenswürdigen Auftraggeber

auflisten. Wenn die Dokumentation für den Dienst oder das Feature, die Sie verwenden, keine Anweisungen zur Auflistung des Dienstes als vertrauenswürdigen Auftraggeber enthält, geben Sie Feedback für die Seite an.

Um eine Servicerolle manuell erstellen zu können, müssen Sie den Dienstauftraggeber für den Dienst kennen, der die Rolle übernehmen wird. Ein Dienstauftraggeber ist eine Kennung, die verwendet wird, um einem Service Berechtigungen zu erteilen. Der Dienstauftraggeber wird durch den Service definiert.

Sie finden den Dienstauftraggeber für einige Dienste, indem Sie Folgendes überprüfen:

1. Öffnen Sie AWS Dienste, die mit IAM funktionieren.
2. Überprüfen Sie, ob der Service in der Spalte Service-linked roles (Serviceverknüpfte Rollen) Yes (Ja) anzeigt.
3. Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.
4. Suchen Sie den Abschnitt „Berechtigungen von serviceverknüpften Rollen“ für diesen Service, um den Dienstauftraggeber anzuzeigen.

Sie können eine Servicerolle manuell mithilfe von AWS CLI Befehlen oder AWS API-Operationenerstellen. Um eine Servicerolle manuell mit der IAM-Konsole zu erstellen, führen Sie die folgenden Aufgaben aus:

1. Erstellen Sie eine IAM-Rolle mit Ihrer Konto-ID. Hängen Sie keine Richtlinie an oder gewähren Sie keine Berechtigungen. Details hierzu finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer.
2. Öffnen Sie die Rolle und bearbeiten Sie die Vertrauensbeziehung. Anstatt dem Konto zu vertrauen, muss die Rolle dem Dienst vertrauen. Aktualisieren Sie zum Beispiel das folgende `Principal`-Element:

   ```
   "Principal": { "AWS": "arn:aws:iam::123456789012:root" }
   ```

   Ändern Sie den Auftraggeber auf den Wert für Ihren Service, wie z. B. IAM.

   ```
   "Principal": { "Service": "iam.amazonaws.com" }
   ```

3. Fügen Sie die Berechtigungen hinzu, die der Dienst benötigt, indem Sie der Rolle Berechtigungsrichtlinien hinzufügen.
4. Kehren Sie zu dem Dienst zurück, der die Berechtigungen benötigt, und verwenden Sie die dokumentierte Methode, um den Dienst über die neue Servicerolle zu informieren.

Fehlersuche bei IAM und Amazon EC2

Verwenden Sie die hier aufgeführten Informationen, um Probleme durch Verweigerung des Zugriffs oder andere Probleme, die beim Arbeiten mit Amazon EC2 und IAM auftreten können, zu beheben.

Themen

Beim Versuch, eine Instance zu starten, sehe ich nicht die Rolle, die ich in der Liste IAM-Rolle der Amazon EC2-Konsole erwartet hatte

Überprüfen Sie, ob Folgendes der Fall ist:

- Wenn Sie als IAM-Benutzer angemeldet sind, stellen Sie sicher, dass Sie über die Berechtigung zum Aufrufen von `ListInstanceProfiles` verfügen. Informationen zu den erforderlichen Berechtigungen für die Arbeit mit Rollen finden Sie unter "Erforderliche Berechtigungen für die Verwendung von Rollen mit Amazon EC2" in Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden. Informationen zum Hinzufügen von Berechtigungen zu einem Benutzer finden Sie unter Verwalten von IAM-Richtlinien.

Wenn Sie Ihre eigenen Berechtigungen nicht ändern können, müssen Sie sich an einen Administrator wenden, der mit IAM arbeiten kann, um Ihre Berechtigungen zu aktualisieren.

- Wenn Sie eine Rolle mithilfe der IAM-CLI oder -API erstellt haben, überprüfen Sie, ob Sie ein Instance-Profil erstellt und die Rolle diesem Instance-Profil hinzugefügt haben. Wenn Sie Ihre Rolle und Ihr Instanceprofil unterschiedlich benennen, wird in der Liste der IAM-Rollen in der Amazon EC2-Konsole auch nicht der richtige Rollenname angezeigt. Die IAM-Rollen-Liste in der Amazon EC2-Konsole listet die Namen der Instanceprofile auf, nicht die Namen der Rollen. Sie müssen den Namen des Instance-Profils auswählen, das die gewünschte Rolle enthält. Weitere Informationen zu Instance-Profilen finden Sie unter Verwenden von Instance-Profilen.

> **Note**
>
> Wenn Sie die IAM-Konsole zum Erstellen von Rollen verwenden, müssen Sie nicht mit Instance-Profilen arbeiten. Für jede Rolle, die Sie in der IAM-Konsole erstellen, wird ein Instance-Profil mit demselben Namen wie die Rolle erstellt, und die Rolle wird automatisch diesem Instance-Profil hinzugefügt. Ein Instance-Profil kann nur eine IAM-Rolle enthalten und dieses Limit kann nicht erhöht werden.

Die Anmeldeinformationen auf meiner Instance beziehen sich auf die falsche Rolle.

Die Rolle im Instance-Profil wurde möglicherweise kürzlich ausgetauscht. Wenn dies der Fall ist, muss Ihre Anwendung auf die nächste automatisch geplante Rotation von Anmeldeinformationen warten. Erst dann sind wieder Anmeldeinformationen für Ihre Rolle verfügbar.

Wenn Sie die Änderung erzwingen möchten, müssen Sie die Zuweisung des Instance-Profils aufheben und dann das Instance-Profil zuweisen, oder Sie beenden Ihre Instance und starten sie neu.

Wenn ich versuche, **`AddRoleToInstanceProfile`** aufzurufen, wird ein **`AccessDenied`**-Fehler angezeigt

Wenn Sie Anforderungen als IAM-Benutzer erstellen, vergewissern Sie sich, dass Sie über die folgenden Berechtigungen verfügen:

- `iam:AddRoleToInstanceProfile` mit der Ressource entsprechend dem ARN des Instance-Profils (z. B. `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`)

Weitere Informationen über die für die Arbeit mit Rollen erforderlichen Berechtigungen finden Sie unter "How Do I Get Started?" in Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden erlauben. Informationen zum Hinzufügen von Berechtigungen zu einem Benutzer finden Sie unter Verwalten von IAM-Richtlinien.

Amazon EC2 Wenn ich versuche, eine Instance mit einer Rolle zu starten, wird ein **`AccessDenied`**-Fehler angezeigt.

Überprüfen Sie, ob Folgendes der Fall ist:

- Starten Sie eine Instance ohne Instance-Profil. Auf diese Weise wird sichergestellt, dass das Problem auf IAM-Rollen für Amazon EC2-Instances beschränkt ist.
- Wenn Sie Anforderungen als IAM-Benutzer erstellen, vergewissern Sie sich, dass Sie über die folgenden Berechtigungen verfügen:
  - `ec2:RunInstances` mit einer Platzhalterressource ("*")
  - `iam:PassRole` mit der Ressource entsprechend dem ARN der Rolle (z. B. `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Rufen Sie die IAM-Aktion `GetInstanceProfile` auf, um sicherzustellen, dass Sie einen gültigen Instance-Profilnamen oder einen gültigen Instance-Profil-ARN verwenden. Weitere Informationen finden Sie unter Using IAM roles with Amazon EC2 instances.
- Rufen Sie die IAM-Aktion `GetInstanceProfile` auf, um sicherzustellen, dass das Instance-Profil eine Rolle hat. Leere Instance-Profile schlagen mit einem `AccessDenied`-Fehler fehl. Weitere Informationen zum Erstellen einer Rolle finden Sie unter Erstellen von IAM-Rollen.

Weitere Informationen über die für die Arbeit mit Rollen erforderlichen Berechtigungen finden Sie unter "How Do I Get Started?" in Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden erlauben. Informationen zum Hinzufügen von Berechtigungen zu einem Benutzer finden Sie unter Verwalten von IAM-Richtlinien.

Ich kann nicht auf die temporären Anmeldeinformationen in meiner EC2-Instance zugreifen.

Um auf die temporären Sicherheitsanmeldeinformationen Ihrer EC2-Instance zuzugreifen, müssen Sie zuerst die IAM-Konsole verwenden, um eine Rolle zu erstellen. Dann starten Sie eine EC2-Instance, die diese Rolle verwendet und untersuchen die laufende Instance. Weitere Informationen finden Sie unter Erste Schritte in Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden.

Wenn Sie immer noch keinen Zugriff auf Ihre temporären Sicherheitsanmeldeinformationen in Ihrer EC2-Instance haben, überprüfen Sie Folgendes:

- Können Sie auf einen anderen Teil des Instance Meta Data Service (IMDS) zugreifen? Falls nicht, überprüfen Sie, ob nicht Firewall-Regeln den Zugriff auf Anfragen an den IMDS blockieren.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- Ist die `iam`-Unterstruktur des IMDS vorhanden? Wenn nicht, überprüfen Sie, ob Ihrer Instance ein IAM-Instance-Profil zugeordnet ist, indem Sie die EC2 `DescribeInstances`-API-Operation aufrufen oder den `aws ec2 describe-instances`-CLI-Befehl verwenden.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Überprüfen Sie das `info`-Dokument in der IAM-Unterstruktur auf Fehler. Wenn ein Fehler vorliegt, finden Sie weitere Informationen unter Was bedeuten die Fehler aus dem `info`-Dokument in der IAM-Unterstruktur?.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

Was bedeuten die Fehler aus dem **`info`**-Dokument in der IAM-Unterstruktur?

Das **`iam/info`**-Dokument gibt Folgendes an: **`"Code":"InstanceProfileNotFound"`**

Ihr IAM-Instance-Profil wurde gelöscht und Amazon EC2 kann keine Anmeldeinformationen mehr für Ihre Instance bereitstellen. Sie müssen ein gültiges Instance-Profil zu Ihrer Amazon EC2-Instance anfügen.

Wenn ein Instance-Profil mit diesem Namen vorhanden ist, prüfen Sie, ob das Instance-Profil nicht gelöscht und ein zweites Profil mit demselben Namen erstellt wurde:

1. Rufen Sie die IAM–`GetInstanceProfile`Operation auf, um die `InstanceProfileId` zu erhalten.
2. Rufen Sie die Amazon EC2–`DescribeInstances`Operation auf, um die `IamInstanceProfileId` für die Instance zu erhalten.
3. Überprüfen Sie, ob die `InstanceProfileId` aus der IAM-Operation mit der `IamInstanceProfileId` aus der Amazon EC2-Operation übereinstimmt.

Wenn die IDs unterschiedlich sind, dann ist das an Ihre Instances angefügte Instance-Profil nicht mehr gültig. Sie müssen ein gültiges Instance-Profil an Ihre Instance anfügen.

Das **`iam/info`**-Dokument gibt an, dass der Vorgang erfolgreich war, gibt aber Folgendes an: **`"Message":"Instance Profile does not contain a role..."`**

Die Rolle wurde mit der IAM-Aktion `RemoveRoleFromInstanceProfile` aus dem Instance-Profil entfernt. Sie können die IAM–`AddRoleToInstanceProfile`-Aktion verwenden, um eine Rolle an das Instance-Profil anzufügen. Ihre Anwendung muss bis zur nächsten geplanten Aktualisierungen warten, um auf die Anmeldeinformationen für die Rolle zuzugreifen.

Wenn Sie die Änderung erzwingen möchten, müssen Sie die Zuweisung des Instance-Profils aufheben und dann das Instance-Profil zuweisen, oder Sie beenden Ihre Instance und starten sie neu.

Das **iam/security-credentials/[role-name]**-Dokument gibt Folgendes an: **"Code":"AssumeRoleUnauthorizedAccess"**

Amazon EC2; verfügt nicht über die Berechtigung, die Rolle zu übernehmen. Die Berechtigung zum Übernehmen der Rolle wird über die an die Rolle angefügte Vertrauensrichtlinie gesteuert, wie im folgenden Beispiel veranschaulicht: Verwenden Sie die IAM–`UpdateAssumeRolePolicy`-API zum Aktualisieren der Vertrauensrichtlinie.

```
{"Version": "2012-10-17","Statement": [{"Effect": "Allow","Principal": {"Service":
 ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

Ihre Anwendung muss bis zur nächsten automatisch geplanten Aktualisierungen warten, um auf die Anmeldeinformationen für die Rolle zuzugreifen.

Wenn Sie die Änderung erzwingen möchten, müssen Sie die Zuweisung des Instance-Profils aufheben und dann das Instance-Profil zuweisen, oder Sie beenden Ihre Instance und starten sie neu.

Fehlerbehebung bei IAM und Amazon S3

Diese Informationen helfen Ihnen bei der Fehlersuche und der Behebung von Problemen, die bei der Arbeit mit Amazon S3 und IAM auftreten können.

Wie gewähre ich anonymen Zugriff auf einen Amazon S3-Bucket?

Verwenden Sie eine Amazon S3-Bucket-Richtlinie mit einem Platzhalter (*) im Element `principal`, sodass beliebige Benutzer auf den Bucket zugreifen können. Mit anonymem Zugriff kann jeder (auch Benutzer ohne AWS-Konto) auf den Bucket zugreifen. Beispiele für Richtlinien finden Sie unter Beispiele für Amazon-S3-Bucket-Richtlinien im Benutzerhandbuch für Amazon Simple Storage Service.

Ich bin als AWS-Konto Root-Benutzer angemeldet. Warum kann ich unter meinem Konto nicht auf einen Amazon S3 S3-Bucket zugreifen?

Möglicherweise haben Sie einen IAM-Benutzer mit Vollzugriff auf IAM und Amazon S3. Wenn der IAM-Benutzer einem Amazon S3 S3-Bucket eine Bucket-Richtlinie zuweist und diese nicht Root-Benutzer des AWS-Kontos als Principal angibt, wird dem Root-Benutzer der Zugriff auf diesen Bucket

verweigert. Als Stammbenutzer können Sie immer noch auf den Bucket zugreifen. Bearbeiten Sie dazu die Bucket-Richtlinie, um einen Stammbenutzer-Zugriff über die Amazon S3-Konsole oder der AWS CLI zuzulassen. Verwenden Sie den folgenden Prinzipal, indem Sie *123456789012* durch die ID des AWS-Konto ersetzen.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Problembehandlung beim SAML 2.0-Verbund mit AWS

Verwenden Sie die hier aufgeführten Informationen, um Probleme zu diagnostizieren und zu beheben, die beim Arbeiten mit SAML 2.0 und Verbund in IAM auftreten können.

Themen

- Fehler: Die Antwort enthält nicht die erforderliche Zielgruppe.
- So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an

Fehler: Ihre Anforderung enthielt eine ungültige SAML-Antwort. Klicken Sie hier, um sich abzumelden.

Dieser Fehler kann auftreten, wenn die SAML-Antwort des Identitätsanbieters kein Attribut mit `Name` mit dem Wert `https://aws.amazon.com/SAML/Attributes/Role` enthält. Das Attribut muss mindestens ein Element `AttributeValue` mit durch Komma getrennten Zeichenfolgen enthalten:

- Der ARN einer Rolle, der der Benutzer zugeordnet werden kann
- Der ARN des SAML-Anbieters

Weitere Informationen finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren. Befolgen Sie die Anleitung unter So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an, um die SAML-Antwort im Browser anzuzeigen.

Fehler: RoleSessionName ist erforderlich in AuthnResponse (Dienst: AWSSecurityTokenService; Statuscode: 400; Fehlercode: InvalidIdentityToken)

Dieser Fehler kann auftreten, wenn die SAML-Antwort des Identitätsanbieters kein Attribut mit `Name` mit dem Wert `https://aws.amazon.com/SAML/Attributes/RoleSessionName` enthält. Der Attributwert ist eine Kennung für den Benutzer. Normalerweise handelt es sich dabei um eine Benutzer-ID oder eine E-Mail-Adresse.

Weitere Informationen finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren. Befolgen Sie die Anleitung unter So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an, um die SAML-Antwort im Browser anzuzeigen.

Fehler: Nicht autorisiert, sts auszuführen: AssumeRole withSAML (Service: AWSSecurityTokenService; Statuscode: 403; Fehlercode:) AccessDenied

Dieser Fehler kann auftreten, wenn die in der SAML-Antwort angegebene IAM-Rolle einen Tippfehler enthält oder nicht vorhanden ist. Achten Sie darauf, den genauen Namen Ihrer Rolle zu verwenden,

da bei Rollennamen die Groß-/Kleinschreibung beachtet wird. Korrigieren Sie den Namen der Rolle in der Konfiguration des SAML-Serviceanbieters.

Sie haben nur dann Zugriff, wenn Ihre Rollenvertrauensrichtlinie die `sts:AssumeRoleWithSAML`-Aktion enthält. Wenn Ihre SAML-Zusicherung für die Verwendung des `PrincipalTag`-Attributs konfiguriert ist, muss Ihre Vertrauensrichtlinie auch die `sts:TagSession`-Aktion enthalten. Weitere Hinweise zu Sitzungs-Tags finden Sie unter Sitzungs-Tags übergeben AWS STS.

Dieser Fehler kann auftreten, wenn Sie nicht `sts:SetSourceIdentity`-Berechtigungen in Ihrer Rollenvertrauensrichtlinie. Wenn Ihre SAML-Zusicherung für die Verwendung des `SourceIdentity`-Attributs konfiguriert ist, muss Ihre Vertrauensrichtlinie auch die `sts:SetSourceIdentity`-Aktion enthalten. Weitere Informationen zu Quellidentität finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

Dieser Fehler kann auch auftreten, wenn die Verbundbenutzer nicht über die Berechtigung zum Annehmen der Rolle verfügen. Die Rolle muss eine Vertrauensrichtlinie enthalten, in der der ARN des IAM-SAML-Identitätsanbieter als `Principal` angegeben ist. Die Rolle enthält zudem Bedingungen, über die gesteuert wird, welche Benutzer die Rolle annehmen können. Stellen Sie sicher, dass Benutzer die Anforderungen dieser Bedingungen erfüllen.

Dieser Fehler kann auch auftreten, wenn die SAML-Antwort nicht `Subject` mit dem Wert `NameID` enthält.

Weitere Informationen finden Sie im Thema über das Einrichten von Berechtigungen in AWS für Verbundbenutzer und SAML-Assertionen für die Authentifizierungsantwort konfigurieren. Befolgen Sie die Anleitung unter So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an, um die SAML-Antwort im Browser anzuzeigen.

Fehler: Die RoleSessionName Eingabe AuthnResponse muss mit [a-zA-Z_0-9+=, .@-] {2,64} übereinstimmen (Service:; Statuscode: 400; Fehlercode:) AWSSecurityTokenService InvalidIdentityToken

Dieser Fehler kann auftreten, wenn der Attributwert `RoleSessionName` zu hoch ist oder ungültige Zeichen enthält. Die maximal zulässige Länge beträgt 64 Zeichen.

Weitere Informationen finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren. Befolgen Sie die Anleitung unter So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an, um die SAML-Antwort im Browser anzuzeigen.

Fehler: Die Quellenidentität muss mit [A-Za-Z_0-9+=, .@-] {2,64} übereinstimmen und darf nicht mit **"aws:"** (Service:; Statuscode: 400; Fehlercode: AWSSecurityTokenService) beginnen InvalidIdentityToken

Dieser Fehler kann auftreten, wenn der Attributwert `sourceIdentity` zu hoch ist oder ungültige Zeichen enthält. Die maximal zulässige Länge beträgt 64 Zeichen. Weitere Informationen zu Quellidentität finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

Weitere Informationen zum Erstellen von SAML-Assertionen finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren. Befolgen Sie die Anleitung unter So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an, um die SAML-Antwort im Browser anzuzeigen.

Fehler: Die Antwortsignatur ist ungültig (Dienst:; Statuscode: 400; Fehlercode: AWSSecurityTokenService) InvalidIdentityToken

Dieser Fehler kann auftreten, wenn Verbundmetadaten des Identitätsanbieters nicht mit den Metadaten des IAM-Identitätsanbieters übereinstimmen. Es kann beispielsweise sein, dass sich die Metadatendatei des Identitätsserviceanbieters geändert hat, um ein abgelaufenes Zertifikat zu aktualisieren. Laden Sie die aktualisierte SAML-Metadatendatei des Identitätsserviceanbieters herunter. Aktualisieren Sie es dann in der AWS Identitätsanbieter-Entität, die Sie in IAM mit dem `aws iam update-saml-provider` plattformübergreifenden CLI-Befehl oder dem `Update-IAMSAMLProvider` PowerShell Cmdlet definieren.

Fehler: Die Rolle konnte nicht übernommen werden: Der Aussteller ist beim angegebenen Anbieter nicht vorhanden (Dienst:; Statuscode: 400; Fehlercode: AWSOpenIdDiscoveryService) AuthSamlInvalidSamlResponseException

Dieser Fehler kann auftreten, wenn der Aussteller in der SAML-Antwort nicht mit dem Aussteller übereinstimmt, der in der Verbundmetadatendatei angegeben wurde. Die Metadatendatei wurde hochgeladen, AWS als Sie den Identitätsanbieter in IAM erstellt haben.

Error: Could not parse metadata.

Dieser Fehler kann auftreten, wenn Ihre Metadatendatei nicht ordnungsgemäß formatiert ist.

Wenn Sie in der einen SAML-Identitätsanbieter erstellen oder verwalten AWS Management Console, müssen Sie das SAML-Metadatendokument von Ihrem Identitätsanbieter abrufen.

Diese Metadatendatei enthält den Namen des Ausstellers, Ablaufinformationen und Schlüssel, die zum Überprüfen der vom IdP empfangenen SAML-Authentifizierung (Zusicherungen) verwendet werden können. Die Metadatendatei muss im UTF-8-Format ohne BOM (Byte Order Mark, Markierung der Bytereihenfolge) codiert sein. Zum Entfernen der BOM können Sie die Datei mithilfe eines Textbearbeitung-Tools wie Notepad++ als UTF-8 codieren.

Das x.509-Zertifikat, das Bestandteil des SAML-Metadatendokuments ist, muss eine Schlüsselgröße von mindestens 1024 Bit verwenden. Außerdem darf das x.509-Zertifikat auch keine wiederholten Erweiterungen enthalten. Sie können Erweiterungen verwenden, diese dürfen jedoch nur einmal im Zertifikat angezeigt werden. Wenn das x.509-Zertifikat keine der beiden Bedingungen erfüllt, schlägt die IdP-Erstellung fehl und gibt den Fehler „Unable to parse metadata“ (Metadaten können nicht analysiert werden) zurück.

Gemäß der Definition im SAML V2.0 Metadata Interoperability Profile Version 1.0 bewertet IAM den Ablauf des X.509-Zertifikats des Metadatendokuments weder, noch ergreift es entsprechende Maßnahmen.

Fehler: Der angegebene Anbieter ist nicht vorhanden.

Dieser Fehler kann auftreten, wenn der Name des Anbieters, den Sie in der SAML-Zusicherung angeben, nicht mit dem des Anbieters übereinstimmt, der in IAM konfiguriert ist. Weitere Informationen zum Anzeigen des Anbieternamens finden Sie unter Erstellen Sie einen SAML-Identitätsanbieter in IAM.

Fehler: Die angeforderte MaxSessionDuration Menge DurationSeconds überschreitet den für diese Rolle festgelegten Wert.

Dieser Fehler kann auftreten, wenn Sie eine Rolle über die API AWS CLI oder übernehmen.

Wenn Sie die Operationen assume-role-with-saml CLI oder AssumeRolewithSAML API verwenden, um eine Rolle anzunehmen, können Sie einen Wert für den Parameter angeben. `DurationSeconds` Sie können einen Wert von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle angeben. Wenn Sie einen Wert höher als diese Einstellung angeben, schlägt die Operation fehl. Wenn Sie beispielsweise eine Sitzungsdauer von zwölf Stunden angeben, Ihr Administrator aber die maximale Sitzungsdauer auf sechs Stunden festgelegt hat, schlägt die Operation fehl. Weitere

Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.

Fehler: Die Antwort enthält nicht die erforderliche Zielgruppe.

Dieser Fehler kann auftreten, wenn in der SAML-Konfiguration eine Diskrepanz zwischen der Zielgruppen-URL und dem Identitätsanbieter besteht. Stellen Sie sicher, dass die ID für die vertrauende Seite Ihres Identitätsanbieters (IDP) exakt mit der Zielgruppen-URL (Entitäts-ID) übereinstimmt, die in der SAML-Konfiguration angegeben ist.

So zeigen Sie eine SAML-Antwort in Ihrem Browser zwecks Fehlerbehebung an

In den folgenden Verfahren wird beschrieben, wie Sie die SAML-Antwort Ihres Dienstanbieters in Ihrem Browser anzeigen, wenn Sie ein Problem im Zusammenhang mit SAML 2.0 beheben.

Öffnen Sie in Ihrem jeweiligen Browser die Seite, auf der Sie das Problem reproduzieren können. Führen Sie dann die Schritte in Ihrem jeweiligen Browser aus:

Themen

Google Chrome

So zeigen Sie eine SAML-Antwort in Chrome an

Diese Schritte wurden mit der Version 106.0.5249.103 (Official Build) (arm64) von Google Chrome getestet. Wenn Sie eine andere Version verwenden, müssen Sie diese Schritte möglicherweise entsprechend anpassen.

1. Drücken Sie F12, um die Developer-Tools-Konsole zu starten.
2. Wählen Sie die Registerkarte Network (Netzwerk) und dann oben links im Fenster Developer Tools (Entwicklertools) die Option Preserve log (Protokoll beibehalten) aus.
3. Reproduzieren Sie das Problem.

4. (Optional) Wenn die Spalte Method (Methode) im Protokollbereich Developer Tools (Entwicklertools) Network (Netzwerk) nicht sichtbar ist, klicken Sie mit der rechten Maustaste auf eine beliebige Spaltenbezeichnung, und wählen Sie Method (Methode) aus, um die Spalte hinzuzufügen.
5. Suchen Sie im Protokollbereich Developer Tools (Entwicklertools) Network (Netzwerk) nach einem SAML-Beitrag. Wählen Sie diese Zeile aus und öffnen Sie dann oben die Registerkarte Payload (Nutzlast). Suchen Sie nach dem Element SAMLResponse, das die kodierte Anforderung enthält. Der zugehörige Wert ist die Base64-kodierte Antwort.

Mozilla Firefox

So zeigen Sie eine SAML-Antwort in Firefox an

Diese Schritte wurden mit der Mozilla-Firefox-Version 105.0.3 (64-bit) getestet. Wenn Sie eine andere Version verwenden, müssen Sie diese Schritte möglicherweise entsprechend anpassen.

1. Drücken Sie F12, um die Web-Developer-Tools-Konsole zu starten.
2. Wählen Sie die Registerkarte Network (Netzwerk).
3. Klicken Sie oben rechts im Fenster Web Developer Tools (Entwicklertools) auf „Options“ (Optionen) (das kleine Zahnradsymbol). Wählen Sie Persist logs (Protokolle beibehalten) aus.
4. Reproduzieren Sie das Problem.
5. (Optional) Wenn die Spalte Method (Methode) im Protokollbereich Web Developer Tools (Webentwicklertools) Network (Netzwerk) nicht sichtbar ist, klicken Sie mit der rechten Maustaste auf eine beliebige Spaltenbezeichnung, und wählen Sie Method (Methode) aus, um die Spalte hinzuzufügen.
6. Suchen Sie in der Tabelle nach einer POST SAML. Wählen Sie diese Zeile aus, rufen Sie dann die Registerkarte Request (Anfrage) auf und suchen Sie das Element SAMLResponse. Der zugehörige Wert ist die Base64-kodierte Antwort.

Apple Safari

So zeigen Sie eine SAML-Antwort in Safari an

Diese Schritte wurden mit Version 16.0 (17614.1.25.9.10, 17614) von Apple Safari getestet. Wenn Sie eine andere Version verwenden, müssen Sie diese Schritte möglicherweise entsprechend anpassen.

1. Aktivieren Sie Web Inspector in Safari. Öffnen Sie das Fenster Preferences (Präferenzen), klicken Sie auf die Registerkarte Advanced (Erweitert) und wählen Sie dann die Option für Show Develop menu in the menu bar (Menü "Entwickeln" in der Menüleiste anzeigen).
2. Nun können Sie Web Inspector öffnen. Wählen Sie in der Menüleiste Develop (Entwickeln) und dann Show Web Inspector (Web Inspector anzeigen) aus.
3. Wählen Sie die Registerkarte Network (Netzwerk).
4. Wählen Sie oben links im Fenster Web Inspector „Options“ (Optionen) aus (das kleine Kreissymbol mit drei horizontalen Linien). Wählen Sie Preserve log (Protokoll beibehalten) aus.
5. (Optional) Wenn die Spalte Method (Methode) im Protokollbereich Web Developer Inspector Network (Netzwerk) nicht sichtbar ist, klicken Sie mit der rechten Maustaste auf eine beliebige Spaltenbezeichnung, und wählen Sie Method (Methode) aus, um die Spalte hinzuzufügen.
6. Reproduzieren Sie das Problem.
7. Suchen Sie in der Tabelle nach einer POST SAML. Wählen Sie diese Zeile aus und rufen Sie die Registerkarte „Header“ auf.
8. Suchen Sie nach dem Element SAMLResponse, das die kodierte Anforderung enthält. Führen Sie einen Bildlauf nach unten durch, um nach `Request Data` mit dem Namen `SAMLResponse` zu suchen. Der zugehörige Wert ist die Base64-kodierte Antwort.

Vorgehensweise bei der Base64-kodierten SAML-Antwort

Wenn Sie das Base64-kodierte SAML-Antwortelement in Ihrem Browser gefunden haben, kopieren Sie es und extrahieren Sie die XML-markierte Antwort mit Ihrem bevorzugten Base64-Decodierungstool.

> **Sicherheitshinweis**
>
> Da die angezeigten SAML-Antwortdaten möglicherweise sensible Sicherheitsdaten enthalten, empfehlen wir Ihnen, keinen Online-Base64-Decoder zu verwenden. Verwenden Sie

stattdessen ein auf Ihrem lokalen Computer installiertes Tool, das Ihre SAML-Daten nicht über das Netzwerk sendet.

Integrierte Option für Windows-Systeme (PowerShell):

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Integrierte Option für MacOS- und Linux-Systeme:

```
$ echo "base64encodedtext" | base64 --decode
```

Überprüfen Sie die Werte in der dekodierten Datei

Überprüfen Sie die Werte in der dekodierten SAML-Antwortdatei.

- Stellen Sie sicher, dass der Wert für das Attribut saml:NameID mit dem Benutzernamen des authentifizierten Benutzers übereinstimmt.
- Überprüfen Sie den Wert für https://aws.amazon.com/SAML/Attributes/Role. Beim ARN- und SAML-Anbieter wird zwischen Groß- und Kleinschreibung unterschieden, und der ARN muss mit der Ressource in Ihrem Konto übereinstimmen.
- Überprüfen Sie den Wert für https://aws.amazon.com/SAML/Attributes/ RoleSession Name. Der Wert muss mit dem Wert in der Anspruchsregel übereinstimmen.
- Wenn Sie den Attributwert für eine E-Mail-Adresse oder einen Kontonamen konfigurieren, stellen Sie sicher, dass die Werte korrekt sind. Die Werte müssen der E-Mail-Adresse oder dem Kontonamen des authentifizierten Benutzers entsprechen.

Suchen Sie nach Fehlern und bestätigen Sie die Konfiguration

Überprüfen Sie, ob die Werte Fehler enthalten, und stellen Sie sicher, dass die folgenden Konfigurationen korrekt sind.

- Die Reklamationsregeln erfüllen die erforderlichen Elemente und alle ARNs sind korrekt. Weitere Informationen finden Sie unter Konfigurieren Sie Ihren SAML 2.0-IdP mit dem Vertrauen der Vertrauensperson und dem Hinzufügen von Ansprüchen.

- Sie haben die neueste Metadatendatei von Ihrem IdP AWS in Ihren SAML-Anbieter hochgeladen. Weitere Informationen finden Sie unter Aktivieren des Zugriffs von SAML 2.0-Verbundbenutzern auf AWS Management Console.
- Sie haben die Vertrauensrichtlinie der IAM-Rolle korrekt konfiguriert. Weitere Informationen finden Sie unter Ändern einer Rolle.

Referenzinformationen für AWS Identity and Access Management

In den Themen in diesem Abschnitt finden Sie ausführliches Referenzmaterial für verschiedene Aspekte von IAM und AWS STS.

Themen

Amazon-Ressourcennamen (ARNs)

Amazon Resource Names (ARNs) identifizieren AWS Ressourcen eindeutig. Wir benötigen einen ARN, wenn Sie eine Ressource für alle eindeutig angeben müssen AWS, z. B. in IAM-Richtlinien, Amazon Relational Database Service (Amazon RDS) -Tags und API-Aufrufen.

ARN-Format

Im Folgenden finden Sie die allgemeinen Formate für ARNs. Die spezifischen Formate hängen von der Ressource ab. Um einen ARN zu verwenden, ersetzen Sie den *kursiv formatierten* Text durch die ressourcenspezifischen Informationen. Beachten Sie, dass die ARNs für einige Ressourcen die Region, die Konto-ID oder sowohl die Region als auch die Konto-ID weglassen.

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

Die Partition, in der sich die Ressource befindet. Eine Partition ist eine Gruppe von Regionen. AWS Jedes AWS Konto ist auf eine Partition beschränkt.

Im Folgenden werden die unterstützten Partitionen angezeigt:

- `aws`- Regionen AWS
- `aws-cn` – China-Regionen
- `aws-us-gov` – AWS GovCloud (US) -Regionen

service

Der Service-Namespace, der das AWS Produkt identifiziert.

region

Der Regionscode. Zum Beispiel `us-east-2` für USA Ost (Ohio). Eine Liste der Regionscodes finden Sie unter Regionale Endpunkte in der Allgemeine AWS-Referenz.

account-id

Die ID des AWS Kontos, dem die Ressource gehört, ohne Bindestriche. z. B. `123456789012`.

resource-type

Der Ressourcentyp. Zum Beispiel `vpc` für eine Virtual Private Cloud (VPC).

resource-id

Die Ressourcen-ID Dies ist der Name der Ressource, die ID der Ressource oder ein Ressourcenpfad. Einige Ressourcen-IDs enthalten eine übergeordnete Ressource (sub-resource-type/parent-resource/sub-resource) oder einen Qualifizierer wie eine Version (resource-type:resource-name:qualifier).

Beispiele

IAM-Benutzer

arn:aws:iam::*123456789012*:user/*johndoe*

SNS-Thema

arn:aws:sns: us-east-1:123456789012: example-sns-topic-name

VPC

arn:aws:ec2:*us-east-1*:*123456789012*:vpc/*vpc-0e9801d129EXAMPLE*

Suchen eines ARN-Formats für eine Ressource

Das genaue Format eines ARN hängt vom Service- und Ressourcentyp ab. Einige Ressourcen-ARNs können einen Pfad, eine Variable oder einen Platzhalter enthalten. Um das ARN-Format für eine bestimmte AWS Ressource nachzuschlagen, öffnen Sie die Service Authorization Reference, öffnen Sie die Seite für den Service und navigieren Sie zur Tabelle mit den Ressourcentypen.

Pfade in ARNs

Ressourcen-ARNs können einen Pfad enthalten. In Amazon S3 ist die Ressourcen-ID beispielsweise ein Objektname mit Schrägstrichen (`/`) zur Bildung eines Pfads. Auch IAM-Benutzernamen und -Gruppennamen können Pfade enthalten. In IAM-Pfaden sind nur alphanumerische Zeichen und die folgenden Zeichen zulässig: Schrägstrich (`/`), Pluszeichen (`+`), Gleichheitszeichen (`=`), Komma (`,`), Punkt (`.`), At-Zeichen (`@`), Unterstrich (`_`) und Bindestrich (`-`).

Verwenden von Platzhaltern in Pfaden

Pfade können ein Platzhalterzeichen enthalten, und zwar ein Sternchen (`*`). Wenn Sie beispielsweise eine IAM-Richtlinie schreiben, können Sie alle IAM-Benutzer angeben, die den Pfad `product_1234` haben, indem Sie einen Platzhalter wie folgt verwenden:

```
arn:aws:iam::123456789012:user/Development/product_1234/*
```

Analog dazu können Sie `user/*` für alle Benutzer oder `group/*` für alle Gruppen angeben. Beispiele:

```
"Resource":"arn:aws:iam::123456789012:user/*"
"Resource":"arn:aws:iam::123456789012:group/*"
```

Das folgende Beispiel zeigt ARNs für einen Amazon-S3-Bucket, in dem der Ressourcenname einen Pfad enthält:

```
arn:aws:s3:::my_corporate_bucket/*
arn:aws:s3:::my_corporate_bucket/Development/*
```

Falsche Verwendung von Platzhaltern

Es ist nicht möglich, einen Platzhalter in dem Teil der ARN zu verwenden, der den Ressourcentyp angibt, z. B. das Wort `user` in einem IAM-ARN. Beispielsweise ist Folgendes nicht zulässig:

```
arn:aws:iam::123456789012:u*   <== not allowed
```

IAM-IDs

IAM verwendet einige verschiedene IDs für Benutzer, Gruppen, Rollen, Richtlinien und Serverzertifikate. In diesem Abschnitt werden die IDs und deren jeweilige Verwendung beschrieben.

Themen

Anzeigenamen und -pfade

Wenn Sie einen Benutzer, eine Rolle, eine Gruppe oder eine Richtlinie erstellen oder ein Serverzertifikat hochladen, geben Sie diesen einen Anzeigenamen. Beispiele hierfür sind Bob, TestApp 1, Developers ManageCredentialsPermissions,, oder ProdServerCert.

Wenn Sie die IAM-API oder AWS Command Line Interface (AWS CLI) verwenden, um IAM-Ressourcen zu erstellen, können Sie einen optionalen Pfad hinzufügen. Sie können einen einzelnen Pfad verwenden oder mehrere Pfade in einer Ordnerstruktur verschachteln. Beispielsweise können Sie den verschachtelten Pfad `/division_abc/subdivision_xyz/product_1234/engineering/` verwenden, um der Organisationsstruktur Ihres Unternehmens zu entsprechen. Sie können dann eine Richtlinie einrichten, die allen Benutzern in diesem Pfad ermöglicht, auf die Richtliniensimulator-API zuzugreifen. Informationen zu dieser Richtlinie finden Sie unter: IAM: Zugriff auf die Richtliniensimulator-API basierend auf dem Benutzerpfad. Informationen darüber, wie ein Anzeigename angegeben werden kann, finden Sie in der Dokumentation zur User API. Weitere Beispiele für die Verwendung von Pfaden finden Sie unter IAM-ARNs.

Wenn Sie Ressourcen erstellen AWS CloudFormation , können Sie einen Pfad für Benutzer, Benutzergruppen und Rollen sowie vom Kunden verwaltete Richtlinien angeben.

Wenn Sie einen Benutzer und eine Benutzergruppe im gleichen Pfad haben, fügt IAM den Benutzer nicht automatisch in diese Benutzergruppe ein. Sie können beispielsweise die Gruppe "Developers" erstellen und als deren Pfad `/division_abc/subdivision_xyz/product_1234/engineering/` angeben. Wenn Sie einen Benutzer namens Bob erstellen und ihm denselben Pfad hinzufügen, bedeutet dies nicht, dass Bob automatisch der Benutzergruppe „Developers“ zugeordnet wird. IAM erzwingt keine Abgrenzung zwischen Benutzern oder Gruppen basierend auf deren Pfade. Benutzer mit verschiedenen Pfaden können dieselben Ressourcen verwenden (dabei wird davon ausgegangen, dass Berechtigungen für diese Ressourcen erteilt wurden). Die Anzahl und Größe der IAM-Ressourcen in einem AWS Konto sind begrenzt. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS.

IAM-ARNs

Die meisten Ressourcen haben einen Anzeigenamen, z. B. ein Benutzer mit dem Namen `Bob` oder eine Gruppe mit dem Namen `Developers`). Allerdings ist es für die Sprache der Berechtigungsrichtlinie erforderlich, dass Sie die Ressource oder Ressourcen im folgenden Amazon-Ressourcenname (ARN)-Format angeben.

```
arn:partition:service:region:account:resource
```

Wobei gilt:

- `partition` identifiziert die Partition, in der sich die Ressource befindet. Für AWS -Standardregionen lautet die Partition `aws`. Wenn Sie Ressourcen in anderen Partitionen haben, lautet die Partition `aws-partitionname`. Die Aufteilung der Ressourcen in der Region China (Peking) lautet beispielsweise `aws-cn`. Es ist nicht möglich, zwischen Konten in verschiedenen Partitionen Zugriff zu delegieren.
- `service`identifiziert das AWS Produkt. IAM-Ressourcen verwenden immer `iam`.
- `region` identifiziert die Region der Ressource an. Für IAM-Ressourcen bleibt dies grundsätzlich leer.
- `account`gibt die AWS-Konto ID ohne Bindestriche an.
- `resource` identifiziert die spezifische Ressource anhand ihres Namens.

Sie können IAM und AWS STS ARNs mit der folgenden Syntax angeben. Der Teil zur Angabe der Region im ARN ist leer, da IAM-Ressourcen globale Ressourcen sind.

Syntax:

```
arn:aws:iam::account:root
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
```

Viele der folgenden Beispiele enthalten Pfade im Ressourcen-Teil des ARN. Pfade können in der AWS Management Console weder erstellt noch bearbeitet werden. Um Pfade verwenden zu können, müssen Sie mit der Ressource arbeiten, indem Sie die AWS API AWS CLI, die oder die Tools für Windows verwenden. PowerShell

Beispiele:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

Die folgenden Beispiele bieten detailliertere Informationen, damit Sie das ARN-Format für verschiedene IAM-Typen und AWS STS Ressourcen besser verstehen.

- Ein IAM-Benutzer im Konto:

> **Note**
>
> Jeder IAM-Benutzername muss eindeutig sein. Beim Benutzernamen wird für den Benutzer nicht zwischen Groß- und Kleinschreibung unterschieden, z. B. während des Anmeldevorgangs, aber wenn Sie ihn in einer Richtlinie oder als Teil eines ARN verwenden, wird zwischen Groß- und Kleinschreibung unterschieden.

```
arn:aws:iam::123456789012:user/JohnDoe
```

- Ein anderer Benutzer mit einem Pfad basierend auf einem Organigramm:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- Einem IAM-Benutzer oder einer IAM-Gruppe:

```
arn:aws:iam::123456789012:group/Developers
```

- Eine IAM-Gruppe mit einem Pfad:

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Eine IAM-Rolle:

```
arn:aws:iam::123456789012:role/S3Access
```

- Eine serviceverknüpfte Rolle:

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Eine Servicerolle:

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Eine verwaltete Richtlinie:

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Ein Instance-Profil, das mit einer Amazon-EC2-Instance verknüpft werden kann:

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Ein Verbundbenutzer in IAM mit dem Namen „Paulo“:

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- Die aktive Sitzung einer Benutzerin mit der Rolle "Accounting-Role" und dem Rollen-Sitzungsnamen "Mary":

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- Stellt die eigene Sitzung des Aufrufers dar, wenn sie als Ressource in einem API-Aufruf verwendet wird, z. B. die AWS STS SetContextAPI, die in der aufrufenden Sitzung ausgeführt wird:

```
arn:aws:sts::123456789012:self
```

- Das dem Benutzer Jorge zugewiesene Gerät für Multifaktor-Authentifizierung:

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Ein Serverzertifikat

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Ein Serverzertifikat mit einem Pfad, der ein Organigramm widerspiegelt:

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Identitätsanbieter (SAML und OIDC):

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- OIDC-Identitätsanbieter mit einem Pfad, der eine Amazon-EKS-OIDC-Identitätsanbieter-URL widerspiegelt:

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

Ein weiterer wichtiger ARN ist der Stammbenutzer-ARN. Obwohl dies keine IAM-Ressource ist, sollten Sie mit dem Format dieses ARN vertraut sein. Er wird häufig im `Principal`-Element einer ressourcenbasierter Richtlinie verwendet.

- Das AWS-Konto zeigt Folgendes an:

```
arn:aws:iam::123456789012:root
```

Das folgende Beispiel zeigt eine Richtlinie, die Sie Richard zuweisen können, damit er die Berechtigung zur Verwaltung seiner eigenen Zugriffsschlüssel erhält. Beachten Sie, dass die Ressource der IAM-Benutzer Richard ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/Richard"
```

```
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Wenn Sie ARNs verwenden, um Ressourcen in einer IAM-Richtlinie zu identifizieren, können Sie Richtlinienvariablen einschließen. Richtlinienvariablen können Platzhalter für Laufzeitinformationen (z. B. den Benutzernamen) als Teil des ARN enthalten. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags

Verwenden von Platzhaltern und Pfaden in ARNs

Sie können Platzhalterzeichen zur Angabe der *Ressource* im ARN angeben, um mehrere Benutzer oder Gruppen oder Richtlinien zu definieren. Wenn Sie beispielsweise alle Benutzer, die auf product_1234 arbeiten, angeben möchten, geben Sie Folgendes an:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Wenn Sie Benutzer haben, deren Namen mit der Zeichenfolge `app_` beginnen, können Sie sich mit dem folgenden ARN auf alle beziehen.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Um alle Benutzer, Benutzergruppen oder Richtlinien in Ihrer anzugeben AWS-Konto, verwenden Sie jeweils einen Platzhalter nach dem `user/group/`, oder einem `policy/` Teil des ARN.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Wenn Sie den folgenden ARN für einen Benutzer `arn:aws:iam::111122223333:user/*` angeben, passt er zu beiden der folgenden Beispiele.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Wenn Sie jedoch den folgenden ARN für einen Benutzer `arn:aws:iam::111122223333:user/division_abc*` angeben, stimmt er mit dem zweiten Beispiel überein, aber nicht mit dem ersten.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Verwenden Sie keinen Platzhalter innerhalb von `user/`, `group/`oder `policy/` im ARN. Beispielsweise lässt IAM folgendes nicht zu:

```
arn:aws:iam::123456789012:u*
```

Example Beispiel für die Verwendung von Pfaden und ARNs für eine projektbasierte Gruppe

Pfade können in der AWS Management Console weder erstellt noch bearbeitet werden. Um Pfade verwenden zu können, müssen Sie mit der Ressource arbeiten, indem Sie die AWS API AWS CLI, die oder die Tools für Windows PowerShell verwenden.

In diesem Beispiel erstellt Jules in der Gruppe „Marketing_Admin“ eine projektbasierte Gruppe im Pfad /marketing/. Jules ordnet Benutzer aus verschiedenen Teilen des Unternehmens der Gruppe zu. Dieses Beispiel veranschaulicht, dass keine Beziehung zwischen dem Pfad und den Gruppen, in denen sich der Benutzer befindet, existiert.

Die Marketinggruppe möchte ein neues Produkt einführen, daher erstellt Jules eine neue Gruppe im Pfad /marketing/ mit dem Namen Widget Launch. Jules weist dann die folgende Richtlinie der Gruppe zu, die der Gruppe den Zugriff auf Objekte auf den Teil vom `example_bucket`, der für diese Markteinführung vorgesehen ist, gewährt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/widget/*"}}
    }
  ]
}
```

Jules fügt dann die Benutzer, die an dieser Markteinführung arbeiten, der Gruppe hinzu. Dies schließt Patricia und Eli aus dem /Marketing/-Pfad ein. Außerdem werden auch Chris und Chloe aus dem Pfad /sales/ sowie Alice und Jim aus dem Pfad /legal/ der Gruppe hinzugefügt.

Eindeutige Bezeichner

Wenn IAM einen Benutzer, eine Gruppe, Rolle, Richtlinie, ein Instance-Profil oder Serverzertifikat erstellt, weist es jeder Ressource eine eindeutige ID zu. Die eindeutige ID sieht wie folgt aus:

`AIDAJQABLZS4A3QDU576Q`

In den meisten Fällen verwenden Sie Anzeigenamen und ARNs, wenn Sie mit IAM-Ressourcen arbeiten. Auf diese Weise müssen Sie die eindeutige ID für eine bestimmte Ressource nicht kennen. Die eindeutige ID kann allerdings in einigen Fällen nützlich sein, wenn die Verwendung von Anzeigenamen unpraktisch ist.

In einem Beispiel werden benutzerfreundliche Namen in Ihrem AWS-Konto wiederverwendet. Innerhalb Ihres Kontos muss ein Anzeigename für einen Benutzer, eine Gruppe, eine Rolle oder eine Richtlinie eindeutig sein. Sie können beispielsweise einen IAM-Benutzer mit dem Namen `John` erstellen. Ihr Unternehmen verwendet Amazon S3 und hat einen Bucket mit Ordnern für jeden Mitarbeiter. IAM-Benutzer `John` ist ein Mitglied einer IAM-Benutzergruppe mit dem Namen `User-S3-Access` mit Berechtigungen, mit denen Benutzer nur auf ihre eigenen Ordner im Bucket zugreifen können. Ein Beispiel für das Erstellen einer identitätsbasierten Richtlinie, die es IAM-Benutzern erlaubt, auf ihr eigenes Bucket-Objekt in S3 unter Verwendung des Anzeigenamens von Benutzern zuzugreifen, finden Sie unter Amazon S3:: Gewährt IAM-Benutzern programmgesteuert und in der Konsole Zugriff auf ihr S3-Stammverzeichnis..

Angenommen, der Mitarbeiter mit dem Namen John verlässt Ihr Unternehmen und Sie löschen den entsprechenden IAM-Benutzer mit Namen `John`. Später jedoch fängt ein anderer Mitarbeiter mit dem Namen John an und Sie erstellen einen neuen IAM-Benutzer mit dem Namen `John`. Sie fügen den neuen IAM-Benutzer mit dem Namen `John` zur bestehenden IAM-Benutzergruppe `User-S3-Access` hinzu. Wenn die Richtlinie, die mit der Benutzergruppe verbunden ist, den

benutzerfreundlichen IAM-Benutzernamen `John` angibt, ermöglicht die Richtlinie dem neuen John den Zugriff auf Informationen, die der ehemalige John hinterlassen hat.

Im Allgemeinen empfehlen wir, dass Sie in Ihren Richtlinien den -ARN für die Ressource anstelle ihrer eindeutigen ID angeben. Allerdings haben IAM-Benutzer eine eindeutige ID, selbst wenn Sie einen neuen IAM-Benutzer mit einem vorher gelöschten Anzeigenamen erstellen. In diesem Beispiel haben der alte IAM-Benutzer `John` und der neue IAM-Benutzer `John` unterschiedliche, eindeutige IDs. Sie können ressourcenbasierte Richtlinien erstellen, die den Zugriff nach einer eindeutigen ID und nicht nur nach dem Benutzernamen gewähren. Dadurch wird die Wahrscheinlichkeit verringert, dass Sie versehentlich Zugriff auf Informationen gewähren, die ein Mitarbeiter nicht erhalten sollte.

Das folgende Beispiel zeigt, wie Sie eindeutige IDs im `Principal`-Element einer ressourcenbasierten Richtlinie angeben können.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

Das folgende Beispiel zeigt, wie Sie mithilfe von `Condition`-Element einer Richtlinie mit globalem Bedingungsschlüssel `aws:userid` eindeutige IDs angeben können.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Ein weiteres Beispiel, in dem Benutzer-IDs nützlich sein können, ist, wenn Sie Ihre eigene Datenbank (oder ein anderes Speichermedium) für IAM-Benutzer- oder -Rolleninformationen pflegen. Die eindeutige ID kann einen eindeutigen Bezeichner für jeden von Ihnen erstellten IAM-Benutzer oder -Rolle bereitstellen. Dies ist der Fall, auch wenn Sie im Laufe der Zeit über IAM-Benutzer oder -Rollen verfügen, die wie im vorherigen Beispiel einen Namen wiederverwenden.

Grundlegendes zu eindeutigen ID-Präfixen

IAM verwendet die folgenden Präfixe, um anzugeben, auf welche Art von Ressource jede eindeutige ID angewendet wird. Präfixe können je nachdem, wann sie erstellt wurden, variieren.

Präfix	Ressourcentyp
ABIA	AWS STS Dienstträger-Token
ACCA	Kontextspezifische Anmeldeinformationen
AGPA	Benutzergruppe
AIDA	IAM-Benutzer
AIPA	Das Amazon EC2-Instance-Profil
AKIA	Zugriffsschlüssel
ANPA	Verwaltete Richtlinie
ANVA	Version in einer verwalteten Richtlinie
APKA	Öffentlicher Schlüssel
AROA	Rolle
ASCA	Zertifikat
ASIA	Temporäre (AWS STS) Zugriffsschlüssel-IDs verwenden diesen Präfix. Sie sind jedoch nur eindeutig, wenn sie gemeinsam mit dem geheimen Zugriffsschlüssel und dem Sitzungstoken verwendet werden.

Abrufen des eindeutigen Bezeichners

Die eindeutige ID für eine -Ressource ist in der IAM-Konsole nicht verfügbar. Um die eindeutige ID zu erhalten, können Sie die folgenden AWS CLI Befehle oder IAM-API-Aufrufe verwenden.

AWS CLI:

- get-caller-identity
- get-group
- get-role
- get-user
- get-policy
- get-instance-profile
- get-server-certificate

IAM-API:

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM und Kontingente AWS STS

AWS Identity and Access Management (IAM) und AWS Security Token Service (STS) haben Kontingente, die die Größe von Objekten begrenzen. Dies wirkt sich darauf aus, wie Sie ein Objekt benennen, die Anzahl der Objekte, die Sie erstellen können, sowie auf die Anzahl der Zeichen, die Sie beim Übergeben eines Objekts verwenden können.

> **Note**
>
> Verwenden Sie den GetAccountSummaryAPI-Vorgang oder den Befehl, um Informationen auf Kontoebene zur IAM-Nutzung und zu den IAM-Kontingenten zu erhalten. get-account-summary AWS CLI

Anforderungen für den IAM-Namen

Für IAM-Namen gelten die folgenden Anforderungen und Einschränkungen:

- Richtliniendokumente können nur die folgenden Unicode-Zeichen enthalten: horizontaler Tabulator (U+0009), Zeilenvorschub (U+000A), Wagenrücklauf (U+000D) sowie Zeichen im Bereich von U +0020 bis U+00FF.
- Die Namen von Benutzern, Gruppen, Rollen, Richtlinien, Instance-Profilen, Serverzertifikaten und Pfaden müssen alphanummerisch sein und dürfen zudem folgende Zeichen enthalten: Plus (+), Gleichheitszeichen (=), Komma (,), Punkt (.), At (@), Unterstrich (_) und Bindestrich (-). Pfadnamen müssen mit einem Slash (/) beginnen und enden.
- Namen von Benutzern, Gruppen, Rollen und Instance-Profilen müssen innerhalb des Kontos eindeutig sein. Diese lassen sich nicht durch Groß- und Kleinschreibung unterscheiden. Sie können zum Beispiel keine Gruppen mit den Namen **`ADMINS`** und **`admins`** erstellen.
- Der Wert der externen ID, den ein Dritter zur Übernahme einer Rolle verwendet, muss mindestens 2 Zeichen und darf höchstens 1.224 Zeichen lang sein. Der Wert muss alphanumerisch sein ohne Leerzeichen. Er kann auch die folgenden Zeichen enthalten: Pluszeichen (+), Gleichheitszeichen (=), Komma (,), Punkt (.), At-Zeichen (@), Doppelpunkt (:), Schrägstrich (/) und Bindestrich (-). Weitere Informationen über die externe ID finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.
- Richtliniennamen für eingebundene Richtlinien müssen für den Benutzer, die Gruppe oder die Rolle, in die sie eingebettet sind, eindeutig sein. Die Namen können alle Zeichen des einfachen lateinischen Alphabets (ASCII) enthalten, mit Ausnahme der folgenden reservierten Zeichen: Schrägstrich (\), umgekehrter Schrägstrich (/), Sternchen (*), Fragezeichen (?) und Leerzeichen. Diese Zeichen sind gemäß RFC 3986, Abschnitt 2.2 reserviert.
- Benutzerpasswörter (Anmeldeprofile) können beliebige ASCII-Zeichen des Basic-Lateinisch-Blocks enthalten.
- AWS-Konto ID-Aliase müssen AWS produktübergreifend eindeutig und gemäß den DNS-Namenskonventionen alphanumerisch sein. Ein Alias muss aus Kleinbuchstaben bestehen, darf nicht mit einem Bindestrich beginnen oder enden, darf keine zwei aufeinanderfolgenden Bindestriche enthalten und darf keine zwölfstellige Zahl sein.

Eine Liste der Basic-Lateinischen ASCII-Zeichen finden Sie in der Library of Congress Basic Latin (ASCII) Codetabelle.

IAM-Objekt-Kontingente

Kontingente, auch als Grenzwerte bezeichnet AWS, sind die Höchstwerte für die Ressourcen, Aktionen und Elemente in Ihrem. AWS-Konto Sie können Ihre IAM-Kontingente mit Service Quotas verwalten.

Eine Liste der IAM-Service-Endpunkte und Service Quotas finden Sie unter AWS Identity and Access Management -Endpunkte und Kontingente im Allgemeine AWS-Referenz.

So fordern Sie eine Kontingenterhöhung an

1. Folgen Sie dem Anmeldeverfahren, das Ihrem Benutzertyp entspricht, wie im Abschnitt So melden Sie sich bei AWS an im AWS -Anmelde-Benutzerhandbuch, um sich bei der AWS Management Console anzumelden.
2. Öffnen Sie die Service Quotas-Konsole.
3. Wählen Sie im Navigationsbereich AWS -Services.
4. Wählen Sie auf der Navigationsleiste die Region US East (N. Virginia). Dann suchen Sie nach **`IAM`**.
5. Wählen Sie AWS Identity and Access Management (IAM), wählen Sie ein Kontingent und folgen Sie den Anweisungen, um eine Kontingentserhöhung zu beantragen.

Weitere Informationen finden Sie unter Anfordern einer Kontingenterhöhung im Service Quotas-Benutzerhandbuch.

Ein Beispiel für das Anfordern einer IAM-Kontingenterhöhung über die Service Quotas-Konsole finden Sie im folgenden Video.

Anfordern einer IAM-Kontingenterhöhung über die Service Quotas-Konsole.

Sie können eine Erhöhung der Standardkontingente für anpassbare IAM-Kontingente anfordern. Anfragen bis zum maximum quota werden automatisch genehmigt und innerhalb weniger Minuten abgeschlossen.

In der folgenden Tabelle sind die Ressourcen aufgeführt, für die Kontingenterhöhungen automatisch genehmigt werden können.

Anpassbare Kontingente für IAM-Ressourcen

Ressource	Standardkontingent	Höchstkontingent
Von Kunden verwaltete Richtlinien pro Konto	1500	5000
Gruppen pro Konto	300	500
Instance-Profile pro Konto	1000	5000
Verwaltete Richtlinien pro Rolle	10	20
Verwaltete Richtlinien pro Benutzer	10	20
Länge der Vertrauensrichtlinie für Rollen	2048 Zeichen	4096 Zeichen
Rollen pro Konto	1000	5000
Serverzertifikate pro Konto	20	1000

IAM Access Analyzer-Kontingente

Eine Liste der Service-Endpunkte und Service Quotas von IAM Access Analyzer finden Sie unter Endpunkte und Kontingente von IAM Access Analyzer in der Allgemeine AWS-Referenz.

Kontingente für IAM Roles Anywhere

Eine Liste der Service-Endpunkte und Service Quotas von IAM Roles Anywhere finden Sie unter Endpunkte und Kontingente von AWS Identity and Access Management Roles Anywhere in der Allgemeine AWS-Referenz.

IAM- und STS-Zeichenlimits

Im Folgenden sind die maximale Zeichenanzahl und die Größenlimits für IAM und AWS STS. Für die folgenden Limits können Sie keine Erhöhung beantragen.

Beschreibung	Limit
Alias für eine AWS-Konto ID	3–63 Zeichen
Für eingebundene Richtlinien	Sie können einem IAM-Benutzer, einer Rolle oder Gruppe beliebig viele Inline-Richtlinien hinzufügen. Die gesamte aggregierte Richtlini engröße (die Gesamtgröße aller eingebund enen Richtlinien) pro Entität darf jedoch die folgenden Grenzwerte nicht überschreiten: • Die Größe der Benutzerrichtlinie darf 2 048 Zeichen nicht überschreiten. • Die Größe der Rollenrichtlinie darf 10 240 Zeichen nicht überschreiten. • Die Größe der Gruppenrichtlinie darf 5 120 Zeichen nicht überschreiten. **Note** IAM berücksichtigt bei der Berechnun g der Größe einer Richtlinie anhand dieser Grenzwerte keine Leerzeichen.
Für verwaltete Richtlinien	• Die Größe jeder verwalteten Richtlinie darf 6 144 Zeichen nicht überschreiten. **Note** IAM berücksichtigt bei der Berechnun g der Größe einer Richtlinie anhand dieses Grenzwerts keine Leerzeichen.
Group name (Gruppenname)	128 Zeichen

Beschreibung	Limit
Instance-Profilnamen	128 Zeichen
Passwort für ein Anmeldeprofil	1–128 Zeichen
Pfad	512 Zeichen
Richtlinienname	128 Zeichen
Rollenname	64 Zeichen **Important** Wenn Sie beabsichtigen, eine Rolle mit der Funktion „Rolle wechseln“ in der zu verwenden AWS Management Console, dann die Kombination `Path` und `RoleName` darf 64 Zeichen nicht überschreiten.

Beschreibung	Limit
Rollensitzungsdauer	12 Stunden Wenn Sie eine Rolle von der API AWS CLI oder übernehmen, können Sie den `duration-seconds` CLI-Parameter oder den `DurationSeconds` API-Parameter verwenden, um eine längere Rollensitzung anzufordern. Sie können einen Wert von 900 Sekunden (15 Minuten) bis zur maximalen Sitzungsdauer für die Rolle angeben, die zwischen 1 und 12 Stunden liegen kann. Wenn Sie keinen Wert für den `DurationSeconds` -Parameter angeben, sind Ihre Sicherheitsanmeldeinformationen eine Stunde lang gültig. IAM-Benutzern, die in der Konsole die Rolle wechseln, wird die maximale Sitzungsdauer oder die verbleibende Zeit in der Sitzung des Benutzers gewährt, je nachdem, was kürzer ist. Die Einstellung der maximalen Sitzungsdauer schränkt die von AWS -Services angenommenen Sitzungen nicht ein. Weitere Informationen dazu, wie Sie den maximalen Wert für Ihre Rolle anzeigen, finden Sie unter Anzeigen der maximalen Sitzungsdauer für eine Rolle.
Rollensitzungsname	64 Zeichen

Beschreibung	Limit
Sitzungsrichtlinien für Rollen	• Die Größe des übergebenen JSON-Richtliniendokuments und aller übergebenen ARN-Zeichen der verwalteten Richtlinie darf zusammen 2 048 Zeichen nicht überschreiten. • Sie können maximal 10 verwaltete Richtlinien-ARNs übergeben, wenn Sie eine Sitzung erstellen. • Sie können nur ein einzelnes JSON-Richtliniendokument übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer programmgesteuert erstellen. • Darüber hinaus werden bei einer AWS Konvertierung die übergebenen Sitzungsrichtlinien und Sitzungs-Tags in ein gepacktes Binärformat komprimiert, für das ein separates Limit gilt. Das `PackedPolicySize` -Antwortelement gibt in Prozent an, wie nah die Richtlinien und Tags für Ihre Anforderung an der oberen Größengrenze liegen. • Wir empfehlen, dass Sie die Sitzungsrichtlinien mithilfe der AWS CLI AWS OR-API übergeben. Dadurch werden der gepackten Richtlinie AWS Management Console möglicherweise zusätzliche Informationen zur Konsolensitzung hinzugefügt.

Beschreibung	Limit
Sitzungs-Tags für Rollen	• Sitzungs-Tags müssen den Grenzwert für Tag-Schlüssel von 128 Zeichen und das Tag-Wert-Limit von 256 Zeichen einhalten. • Sie können bis zu 50 Sitzungs-Tags übergeben. • Bei einer AWS Konvertierung werden die übergebenen Sitzungsrichtlinien und Sitzungs-Tags in ein gepacktes Binärform at komprimiert, für das ein separates Limit gilt. Sie können Sitzungs-Tags mithilfe der AWS CLI oder AWS -API übergeben. Das `PackedPolicySize` -Antwortelement gibt in Prozent an, wie nah die Richtlinien und Tags für Ihre Anforderung an der oberen Größengrenze liegen.
SAML-Authentifizierungsantwort base64-kodiert	100 000 Zeichen Dieses Zeichenlimit gilt für `assume-role-with-saml` CLI oder `AssumeRoleWithSAML` API-Vorgang.
Tag-Schlüssel	128 Zeichen Dieses Zeichenlimit gilt für Tags auf IAM-Ressourcen und Sitzungs-Tags.
Tag-Wert	256 Zeichen Dieses Zeichenlimit gilt für Tags auf IAM-Ressourcen und Sitzungs-Tags. Tag-Werte können leer sein, was bedeutet, Tag-Werte können eine Länge von 0 Zeichen haben.

Beschreibung	Limit
Eindeutige, von IAM erstellte IDs	128 Zeichen. Zum Beispiel: • Benutzer-IDs, die mit `AIDA` beginnen • Gruppen-IDs, die mit `AGPA` beginnen • Rollen-IDs, die mit `AROA` beginnen • Verwaltete Richtlinien-IDs, die mit `ANPA` beginnen • Serverzertifikat-IDs, die mit `ASCA` beginnen **Note** Dies ist keine vollständige Liste und auch keine Garantie dafür, dass IDs eines bestimmten Typs nur mit der angegebenen Buchstabenkombination beginnen.
Benutzername	64 Zeichen

Schnittstellen-VPC-Endpunkte

Wenn Sie Amazon Virtual Private Cloud (Amazon VPC) zum Hosten Ihrer AWS Ressourcen verwenden, können Sie eine private Verbindung zwischen Ihrer VPC und AWS Security Token Service ()AWS STS herstellen. Sie können diese Verbindung verwenden, AWS STS um mit Ihren Ressourcen in Ihrer VPC zu kommunizieren, ohne das öffentliche Internet nutzen zu müssen.

Amazon VPC ist ein AWS Service, mit dem Sie AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk starten können. Mit einer VPC haben Sie die Kontrolle über Ihre Netzwerkeinstellungen, wie IP-Adressbereich, Subnetze, Routing-Tabellen und Netzwerk-Gateways. Um Ihre VPC zu verbinden AWS STS, definieren Sie einen VPC-Schnittstellen-Endpunkt für. AWS STS Der Endpunkt bietet zuverlässige, skalierbare Konnektivität, AWS STS ohne dass ein Internet-Gateway, eine NAT-Instanz (Network Address Translation) oder eine VPN-Verbindung erforderlich ist. Weitere Informationen finden Sie unter Was ist Amazon VPC? im Amazon-VPC-Leitfaden.

Schnittstellen-VPC-Endpunkte basieren auf AWS PrivateLink einer AWS Technologie, die die private Kommunikation zwischen AWS Diensten über eine elastic network interface mit privaten IP-Adressen ermöglicht. Weitere Informationen finden Sie unter Dienste AWS PrivateLink . AWS

Die folgenden Informationen richten sich an Benutzer von Amazon VPC. Weitere Informationen finden Sie unter Erste Schritte mit IPv4 für Amazon VPC im Amazon-VPC-Leitfaden.

Verfügbarkeit

AWS STS unterstützt derzeit VPC-Endpunkte in den folgenden Regionen:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Africa (Cape Town)
- Asien-Pazifik (Hongkong)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Jakarta)
- Asien-Pazifik (Melbourne)
- Asien-Pazifik (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Canada (Central)
- Kanada West (Calgary)
- China (Peking)
- China (Ningxia)
- Europe (Frankfurt)
- Europa (Irland)

- Europa (London)
- Europa (Milan)
- Europa (Paris)
- Europa (Spain)
- Europa (Stockholm)
- Europa (Zürich)
- Israel (Tel Aviv)
- Naher Osten (Bahrain)
- Naher Osten (VAE)
- Südamerika (São Paulo)
- AWS GovCloud (US-Ost)
- AWS GovCloud (US-West)

Erstellen Sie einen VPC-Endpunkt für AWS STS

Um mit der Verwendung AWS STS mit Ihrer VPC zu beginnen, erstellen Sie einen VPC-Schnittstellen-Endpunkt für. AWS STS Weitere Informationen finden Sie unter Zugreifen auf einen AWS Service über einen Schnittstellen-VPC-Endpunkt im Amazon VPC-Benutzerhandbuch.

Nachdem Sie den VPC-Endpunkt erstellt haben, müssen Sie den entsprechenden regionalen Endpunkt verwenden, um Ihre AWS STS Anfragen zu senden. AWS STS empfiehlt, dass Sie beide `setEndpoint` Methoden `setRegion` und verwenden, um Aufrufe an einen regionalen Endpunkt zu tätigen. Sie können die `setRegion`-Methode allein für manuell aktivierte Regionen, z. B. Asien-Pazifik (Hongkong), verwenden. In diesem Fall werden die Aufrufe an den regionalen STS-Endpunkt weitergeleitet. Informationen zum manuellen Aktivieren einer Region finden Sie unter Verwalten von AWS -Regionen in der Allgemeine AWS-Referenz. Wenn Sie die `setRegion`-Methode allein für standardmäßig aktivierte Regionen verwenden, werden die Aufrufe an den globalen Endpunkt von `https://sts.amazonaws.com` weitergeleitet.

Wenn Sie regionale Endpunkte verwenden, AWS STS ruft andere AWS Dienste entweder über öffentliche Endpunkte oder VPC-Endpunkte mit privater Schnittstelle auf, je nachdem, welche verwendet werden. Nehmen wir beispielsweise an, Sie haben einen VPC-Schnittstellen-Endpunkt für Ressourcen erstellt AWS STS und bereits temporäre Anmeldeinformationen AWS STS von Ressourcen angefordert, die sich in Ihrer VPC befinden. In diesem Fall passieren diese Anmeldeinformationen den Schnittstellen-VPC-Endpunkt standardmäßig. Weitere Informationen zum

Stellen regionaler Anfragen mithilfe von finden Sie AWS STS unter. Verwaltung AWS STS in einem AWS-Region

AWS Dienste, die mit IAM funktionieren

Die unten aufgeführten AWS Dienste sind alphabetisch gruppiert und enthalten Informationen darüber, welche IAM-Funktionen sie unterstützen:

- Dienst — Sie können den Namen eines Dienstes wählen, um die AWS Dokumentation zur IAM-Autorisierung und zum Zugriff für diesen Dienst anzuzeigen.
- Aktionen – Sie können einzelne Aktionen in einer Richtlinie angeben. Wenn der Service dieses Feature nicht unterstützt, wird All actions (Alle Aktionen) im visuellen Editor ausgewählt. In einem JSON-Richtliniendokument müssen Sie `*` im `Action`-Element verwenden. Eine Liste der Aktionen in den einzelnen Diensten finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.
- Berechtigungen auf Ressourcenebene – Sie können mit ARNs einzelne Ressourcen in der Richtlinie angeben. Wenn der Service dieses Feature nicht unterstützt, wird All resources (Alle Ressourcen) im visuellen Richtlinieneditor ausgewählt. In einem JSON-Richtliniendokument müssen Sie `*` im `Resource`-Element verwenden. Einige Aktionen, wie z. B. `List*`-Aktionen, unterstützen die Angabe eines ARN nicht, da sie darauf ausgelegt sind, mehrere Ressourcen zurückzugeben. Wenn ein Service dieses Feature für einige Ressourcen unterstützt, für andere aber nicht, wird in der Tabelle durch Partial darauf hingewiesen. Weitere Informationen dazu finden Sie in der Dokumentation zu dem jeweiligen Service.
- Ressourcenbasierte Richtlinien – Sie können ressourcenbasierte Richtlinien an eine Ressource innerhalb des Service anfügen. Ressourcenbasierte Richtlinien enthalten ein `Principal`-Element zur Angabe der IAM-Identitäten, die auf diese Ressource zugreifen können. Weitere Informationen finden Sie unter Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien.
- ABAC (Authorisierung basierend auf Tags) – Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Bedingungsschlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* oder `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 zur Definition von Berechtigungen auf der Basis von Attributen wie Tags finden Sie unter Wofür ist ABAC? AWS. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden.

- Temporäre Anmeldeinformationen — Sie können kurzfristige Anmeldeinformationen verwenden, die Sie erhalten, wenn Sie sich mit IAM Identity Center anmelden, die Rollen in der Konsole wechseln oder die Sie mithilfe AWS STS der AWS API AWS CLI oder generieren. Sie können nur auf Services mit dem Wert Nein zugreifen, während Sie Ihre langfristigen IAM-Benutzeranmeldeinformationen verwenden. Dies beinhaltet einen Benutzernamen und ein Passwort oder Ihre Benutzerzugriffsschlüssel. Weitere Informationen finden Sie unter Temporäre IAM Sicherheitsanmeldeinformationen.
- Serviceverknüpfte Rollen – Eine serviceverknüpfte Rolle ist ein spezieller Typ von Servicerolle, die dem Service die Berechtigung erteilt, in Ihrem Namen auf Ressourcen in anderen Services zuzugreifen. Wählen Sie den Link Ja oder Teilweise, um die Dokumentation für Services anzuzeigen, die diese Rollen unterstützen. In dieser Spalte wird nicht angegeben, ob der Service Standardservicerollen verwendet. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen.
- Weitere Informationen – Wenn ein Service ein Feature nicht vollständig unterstützt, finden Sie in den Fußnoten Informationen zu eventuellen Einschränkungen und Links zu verwandten Informationen.

Services, die mit IAM funktionieren

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Account Management	Ja	Ja	Nein	Nein	Ja	Nein
AWS Activate Console	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Amplify Admin.	Ja	Ja	Nein	Nein	Ja	Nein
AWS Amplify	Ja	Ja	Nein	Teilweise	Ja	Nein
AWS Amplify UI Builder	Ja	Ja	Nein	Ja	Ja	Nein
Apache Kafka-APIs für Amazon MSK-Cluster	Ja	Ja	Nein	Nein	Ja	Nein
Amazon API Gateway	Ja	Ja	Ja	Nein	Ja	Ja
Amazon API Gateway-Management	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon API Gateway-Management V2	Ja	Ja	Nein	Ja	Ja	Nein
AWS App2Container	Ja	Nein	Nein	Nein	Ja	Nein
AWS AppConfig	Ja	Ja	Nein	Ja	Ja	Nein
AWS AppFabric	Ja	Ja	Nein	Ja	Ja	Nein
Amazon AppFlow	Ja	Ja	Nein	Ja	Ja	Nein
Amazon AppIntegrations	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Application Auto Scaling	Ja	Ja	Nein	Ja	Ja	Ja
AWS Cost Profiler für Anwendungen	Ja	Nein	Nein	Nein	Ja	Nein
AWS Arsenal für Anwendungserkennung	Ja	Nein	Nein	Nein	Ja	Nein
AWS Application Discovery Service	Ja	Nein	Nein	Nein	Ja	Ja
AWS Application Migration Service	Ja	Ja	Nein	Ja	Ja	Ja
AWS Service zur Transformation von Anwendungen	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS App Mesh	Ja	Ja	Nein	Ja	Ja	Ja
AWS App Mesh Vorversion	Ja	Ja	Nein	Nein	Ja	Ja
AWS App Runner	Ja	Ja	Nein	Ja	Ja	Ja
Amazon AppStream 2.0	Ja	Ja	Nein	Ja	Ja	Nein
AWS AppSync	Ja	Ja	Nein	Ja	Ja	Nein
AWS Artifact	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Athena	Ja	Ja	Nein	Ja	Ja	Nein
AWS Audit Manager	Ja	Ja	Nein	Ja	Ja	Ja
AWS Auto Scaling	Ja	Nein	Nein	Nein	Ja	Ja
AWS B2B-Datenaustausch	Ja	Ja	Nein	Ja	Ja	Nein
AWS Backup	Ja	Ja	Ja	Ja	Ja	Ja
AWS Backup Tor	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Backup Speicher	Ja	Nein	Nein	Nein	Ja	Nein
AWS Batch	Ja	Teilweise	Nein	Ja	Ja	Ja
Amazon Bedrock	Ja	Ja	Nein	Ja	Ja	Nein
AWS Billing and Cost Management	Ja	Nein	Nein	Nein	Ja	Ja
AWS Billing and Cost Management Datenexporte	Ja	Ja	Nein	Ja	Ja	Nein
AWS Billing Conductor	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Braket	Ja	Ja	Nein	Ja	Ja	Ja
AWS Budget-Service	Ja	Ja	Nein	Nein	Nein	Nein
AWS BugBust	Ja	Ja	Nein	Ja	Ja	Ja
AWS Certificate Manager (ACM)	Ja	Ja	Nein	Ja	Ja	Ja
AWS Chatbot	Ja	Ja	Nein	Nein	Ja	Ja
Amazon Chime	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Clean Rooms	Ja	Ja	Nein	Ja	Ja	Nein
AWS Clean Rooms ML	Ja	Ja	Nein	Ja	Ja	Nein
AWS Client VPN	Ja	Ja	Nein	Nein	Ja	Ja
AWS Cloud9	Ja	Ja	Ja	Ja	Ja	Ja
AWS Cloud Steuer-API	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Cloud Directory	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS CloudFormation	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudFront	Ja	Ja	Nein	Teilweise	Ja	Teilweise (Informationen)
Amazon CloudFront KeyValueStore	Ja	Ja	Nein	Nein	Ja	Nein
AWS CloudHSM	Ja	Ja	Nein	Ja	Ja	Ja
AWS Cloud Map	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudSearch	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS CloudShell	Ja	Ja	Nein	Nein	Ja	Nein
AWS CloudTrail	Ja	Ja	Teilweise (Informationen)	Teilweise (Informationen)	Ja	Ja
AWS CloudTrail Daten	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudWatch	Ja	Ja	Nein	Ja	Ja	Teilweise (Informationen)
Einblicke in CloudWatch Amazon-Anwendungen	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
CloudWatch Amazon-Anwendungssignale	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudWatch offenbar	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudWatch Internetmonitor	Ja	Ja	Nein	Ja	Ja	Nein
CloudWatch Amazon-Protokolle	Ja	Ja	Ja	Teilweise	Ja	Ja
Amazon CloudWatch Netzwerkmonitor	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudWatch Observability Access Manager	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon CloudWatch RUM	Ja	Ja	Nein	Ja	Ja	Nein
Amazon CloudWatch Synthetics	Ja	Ja	Nein	Ja	Ja	Nein
AWS CodeArtifact	Ja	Ja	Ja	Ja	Ja	Nein
AWS CodeBuild	Ja	Ja	Ja (Informationen)	Teilweise (Informationen)	Ja	Nein
Amazon CodeCatalyst	Ja	Ja	Nein	Ja	Ja	Ja
AWS CodeCommit	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS CodeConnections	Ja	Ja	Nein	Ja	Ja	Nein
AWS CodeDeploy	Ja	Ja	Nein	Ja	Ja	Nein
AWS CodeDeploy Dienst für sichere Host-Befehle	Ja	Nein	Nein	Nein	Ja	Nein
Amazon CodeGuru Profiler	Ja	Ja	Nein	Ja	Ja	Ja
CodeGuru Amazon-Rezensent	Ja	Ja	Nein	Ja	Ja	Ja
CodeGuru Amazon-Sicherheit	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS CodePipeline	Ja	Teilweise	Nein	Ja	Ja	Nein
AWS CodeStar	Ja	Teilweise	Nein	Ja	Ja	Nein
AWS CodeStar Verbindungen	Ja	Ja	Nein	Ja	Ja	Ja
AWS CodeStar Benachrichtigungen	Ja	Ja	Nein	Ja	Ja	Ja
Amazon CodeWhisperer	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Cognito	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Cognito Sync	Ja	Ja	Nein	Nein	Ja	Ja
Amazon-Cognito-Benutzerpools	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Comprehend	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Comprehend Medical	Ja	Nein	Nein	Nein	Ja	Nein
AWS Compute Optimizer	Ja	Nein	Nein	Nein	Ja	Ja
AWS Config	Ja	Teilweise (Informationen)	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Connect	Ja	Ja	Nein	Ja	Ja	Ja
Amazon-Connect-Fälle	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Connect Customer Profiles	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Connect Ausgehende Kommunikation mit hohem Volumen	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Connect Voice ID	Ja	Ja	Nein	Ja	Ja	Nein
AWS Console Mobile Application	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Konsolidierte Fakturierung	Ja	Nein	Nein	Nein	Ja	Nein
AWS Katalog kontrollieren	Ja	Ja	Nein	Nein	Ja	Nein
AWS Control Tower	Ja	Ja	Nein	Nein	Ja	Nein
AWS Cost and Usage Report	Ja	Ja	Nein	Nein	Ja	Nein
AWS Cost Explorer	Ja	Ja	Nein	Ja	Ja	Nein
AWS Cost Optimization Hub	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Customer Verification Service	Ja	Nein	Nein	Nein	Ja	Nein
AWS Database Migration Service	Ja	Ja	Nein (Informationen)	Ja	Ja	Ja
Database Query Metadata Service	Ja	Nein	Nein	Nein	Ja	Nein
AWS Data Exchange	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Data Lifecycle Manager	Ja	Ja	Nein	Ja	Ja	Nein
AWS Data Pipeline	Ja	Ja	Nein	Teilweise	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS DataSync	Ja	Ja	Nein	Ja	Ja	Ja
Amazon DataZone	Ja	Nein	Nein	Nein	Ja	Nein
AWS Deadline Cloud	Ja	Ja	Nein	Ja	Ja	Nein
AWS DeepComposer	Ja	Ja	Nein	Ja	Ja	Nein
AWS DeepRacer	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Detective	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Device Farm	Ja	Ja	Nein	Ja	Ja	Ja
DevOpsAmazon-Guru	Ja	Ja	Nein	Nein	Ja	Ja
AWS Diagnosetools	Ja	Ja	Nein	Ja	Ja	Nein
AWS Direct Connect	Ja	Ja	Nein	Ja	Ja	Ja
AWS Directory Service	Ja	Ja	Nein	Ja	Ja	Nein
Elastische Amazon DocumentDB-Cluster	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon DynamoDB Accelerator (DAX)	Ja	Ja	Nein	Nein	Ja	Ja
Amazon-DynamoDB	Ja	Ja	Ja	Nein	Ja	Nein
Amazon Elastic Compute Cloud (Amazon EC2)	Ja	Teilweise	Nein	Ja	Ja	Teilweise (Informationen)
Amazon EC2 Auto Scaling	Ja	Ja	Nein	Ja	Ja	Ja
EC2 Image Builder	Ja	Ja	Nein	Ja	Ja	Ja
Amazon EC2 Instance Connect	Ja	Ja	Nein	Nein	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon ElastiCache	Ja	Ja	Nein	Ja	Ja	Ja
AWS Elastic Beanstalk	Ja	Teilweise	Nein	Ja	Ja	Ja
Amazon Elastic Block Store (Amazon EBS)	Ja	Teilweise	Nein	Ja	Ja	Nein
Amazon Elastic Container Registry (Amazon ECR)	Ja	Ja	Ja	Ja	Ja	Ja
Amazon Elastic Container Registry Öffentlich (Amazon ECR öffentlich)	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Elastic Container Service (Amazon ECS)	Ja	Teilweise (Informationen)	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Elastic Disaster Recovery	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Elastic File System (Amazon EFS)	Ja	Ja	Ja	Teilweise	Ja	Ja
Amazon Elastic Inference	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Elastic Kubernetes Service (Amazon EKS)	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Elastic Kubernetes Service (Amazon EKS) Auth	Ja	Ja	Nein	Nein	Ja	Nein
AWS Elastic Load Balancing	Ja	Teilweise	Nein	Teilweise	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Elastic Transcoder	Ja	Ja	Nein	Nein	Ja	Nein
AWS Elemental Appliances und Software-Aktivierungsservice	Ja	Ja	Nein	Ja	Ja	Nein
AWS Geräte und Software von Elementar	Ja	Ja	Nein	Ja	Ja	Nein
AWS Elemental MediaConnect	Ja	Ja	Nein	Nein	Ja	Ja
AWS Elemental MediaConvert	Ja	Ja	Nein	Ja	Ja	Nein
AWS Elemental MediaLive	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Elemental MediaPackage	Ja	Ja	Nein	Ja	Ja	Teilweise (Informationen)
AWS Elemental MediaPackage V2	Ja	Ja	Nein	Ja	Ja	Nein
AWS Elemental MediaPackage VOD	Ja	Ja	Nein	Ja	Ja	Teilweise (Informationen)
AWS Elemental MediaStore	Ja	Ja	Ja	Ja	Ja	Nein
AWS Elemental MediaTailor	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Elementare Stützkoffer	Ja	Nein	Nein	Nein	Ja	Nein
AWS Inhalte zur elementaren Support	Ja	Nein	Nein	Nein	Ja	Nein
Amazon EMR	Ja	Ja	Nein	Ja	Ja	Ja
Amazon EMR in EKS	Ja	Ja	Nein	Ja	Ja	Ja
Amazon EMR Serverless	Ja	Ja	Nein	Ja	Ja	Ja
AWS Auflösung der Entität	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon EventBridge	Ja	Ja	Ja	Ja	Ja	Nein
EventBridge Amazon-Pfeifen	Ja	Ja	Nein	Ja	Ja	Nein
Amazon EventBridge Scheduler	Ja	Ja	Nein	Ja	Ja	Nein
EventBridge Amazon-Schemas	Ja	Ja	Ja	Ja	Ja	Nein
AWS Service zur Fehlerinjektion	Ja	Ja	Nein	Ja	Ja	Ja
Amazon FinSpace	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon FinSpace API	Ja	Ja	Nein	Nein	Ja	Nein
AWS Firewall Manager	Ja	Ja	Nein	Ja	Ja	Teilweise
Fleet Hub for AWS IoT Device Management	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Forecast	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Fraud Detector	Ja	Ja	Nein	Ja	Ja	Nein
FreeRTOS	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Kostenloses Kontingent	Ja	Nein	Nein	Nein	Ja	Nein
Amazon FSx	Ja	Ja	Nein	Ja	Ja	Ja
Amazon GameLift	Ja	Ja	Nein	Ja	Ja	Nein
AWS Global Accelerator	Ja	Ja	Nein	Ja	Ja	Ja
AWS Glue	Ja	Ja	Ja	Teilweise	Ja	Nein
AWS Glue DataBrew	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Ground Station	Ja	Ja	Nein	Ja	Ja	Ja
Amazon-Ground-Truth-Kennzeichnung	Ja	Nein	Nein	Nein	Ja	Nein
Amazon GuardDuty	Ja	Ja	Nein	Ja	Ja	Ja
AWS Health APIs und Benachrichtigungen	Ja	Ja	Nein	Nein	Ja	Nein
AWS HealthImaging	Ja	Ja	Nein	Ja	Ja	Nein
AWS HealthLake	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS HealthOmics	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Honeycode	Ja	Ja	Nein	Nein	Ja	Nein
AWS IAM Identity Center	Ja	Ja	Nein	Teilweise	Ja	Ja
IAM Identity Center-Verzeichnis	Ja	Nein	Nein	Nein	Ja	Nein
IAM Identity Center-Identitätsspeicher	Ja	Ja	Nein	Nein	Ja	Nein
IAM-Identity-Center-OIDC-Service	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Identity and Access Management (IAM)	Ja	Ja	Teilweise (Informationen)	Teilweise (Informationen)	Teilweise (Informationen)	Nein
AWS Identity and Access Management Access Analyzer	Ja	Ja	Nein	Ja	Ja	Teilweise
AWS Identity and Access Management Rollen überall	Ja	Ja	Nein	Ja	Ja	Ja
AWS Authentifizierung im Identitätsspeicher	Ja	Nein	Nein	Nein	Ja	Nein
AWS Identity-Synchronisierung	Ja	Ja	Nein	Nein	Ja	Nein
AWS Import/Export	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Inspector	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Inspector Classic	Ja	Nein	Nein	Nein	Ja	Ja
Amazon InspectorScan	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Interactive Video Service	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Interactive Video Service Chat	Ja	Ja	Nein	Ja	Ja	Nein
AWS Invoicing	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS IoT 1-Click	Ja	Ja	Nein	Ja	Ja	Nein
AWS IoT Analytics	Ja	Ja	Nein	Ja	Ja	Nein
AWS IoT	Ja	Ja	Teilweise (Informationen)	Ja	Ja	Nein
AWS IoT Core Geräteberater	Ja	Ja	Nein	Ja	Ja	Nein
AWS IoT Gerätetester	Ja	Nein	Nein	Nein	Ja	Nein
AWS IoT Events	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS IoT FleetWise	Ja	Ja	Nein	Ja	Ja	Nein
AWS IoT Greengrass	Ja	Ja	Nein	Ja	Ja	Nein
AWS IoT Greengrass V2	Ja	Ja	Nein	Teilweise	Ja	Nein
AWS IoT Jobs DataPlane	Ja	Ja	Nein	Nein	Ja	Nein
AWS IoT RoboRunner	Ja	Ja	Nein	Nein	Ja	Nein
AWS IoT SiteWise	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS IoT TwinMaker	Ja	Ja	Nein	Ja	Ja	Ja
AWS IoT Wireless	Ja	Ja	Nein	Ja	Ja	Nein
AWS IQ	Ja	Ja	Nein	Nein	Ja	Ja
AWS IQ-Berechtigungen	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Kendra	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Kendra Intelligent Ranking	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Key Management Service (AWS KMS)	Ja	Ja	Ja	Ja	Ja	Ja
Amazon Keyspaces (für Apache Cassandra)	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Managed Service für Apache Flink	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Managed Service für Apache Flink V2	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Data Firehose	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Kinesis Data Streams	Ja	Ja	Ja	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Kinesis Video Streams	Ja	Ja	Nein	Ja	Ja	Nein
AWS Lake Formation	Ja	Nein	Nein	Nein	Ja	Ja
AWS Lambda	Ja	Ja	Ja	Teilweise (Informationen)	Ja	Teilweise (Informationen)
AWS Launch Wizard	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Lex	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Lex V2	Ja	Ja	Ja	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS License Manager	Ja	Ja	Nein	Ja	Ja	Ja
AWS License Manager Linux-Abonnement-Manager	Ja	Nein	Nein	Nein	Ja	Nein
AWS License Manager Benutzerabonnements	Ja	Nein	Nein	Nein	Ja	Ja
Amazon Lightsail	Ja	Teilweise (Informationen)	Nein	Teilweise (Informationen)	Ja	Ja
Amazon Location Service	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Lookout für Equipment	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Lookout for Metrics	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Lookout for Vision	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Machine Learning	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Macie	Ja	Ja	Nein	Ja	Ja	Ja
AWS Mainframe Modernization	Ja	Ja	Nein	Ja	Ja	Ja
AWS Mainframe Modernization Testen von Anwendungen	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Managed Blockchain	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Managed Blockchain Query	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Managed Grafana	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Managed Service for Prometheus	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Managed Streaming for Apache Kafka (MSK)	Ja	Ja	Teilweise (Informationen)	Ja	Ja	Ja
Amazon Managed Streaming for Kafka Connect	Ja	Ja	Nein	Nein	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Managed Workflows for Apache Airflow	Ja	Ja	Nein	Ja	Ja	Nein
AWS Marketplace	Ja	Nein	Nein	Nein	Ja	Ja
AWS Marketplace Katalog	Ja	Ja	Nein	Ja	Ja	Nein
AWS Marketplace Commerce Analytics	Ja	Nein	Nein	Nein	Nein	Nein
AWS Marketplace - Deployment-Service	Ja	Ja	Nein	Ja	Ja	Nein
AWS Marketplace Erkennung	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Marketplace Entitlement Service	Ja	Nein	Nein	Nein	Ja	Nein
AWS Marketplace Service zur Image-Erstellung	Ja	Nein	Nein	Nein	Ja	Nein
AWS Marketplace Management Portal	Ja	Nein	Nein	Nein	Ja	Nein
AWS Marketplace Metering Service	Ja	Nein	Nein	Nein	Ja	Nein
AWS Marketplace Privater Marketplace	Ja	Nein	Nein	Nein	Ja	Nein
AWS Marketplace Integration der Beschaffungssysteme	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Marketplace Berichterstattung für Verkäufer	 Ja	 Ja	 Nein	 Nein	 Ja	 Nein
AWS Marketplace Einblicke in Lieferanten	 Ja	 Ja	 Nein	 Ja	 Ja	 Nein
Amazon Mechanical Turk	 Ja	 Nein	 Nein	 Nein	 Ja	 Nein
Amazon MediaImport	 Ja	 Nein	 Nein	 Nein	 Nein	 Nein
Amazon MemoryDB for Redis	 Ja	 Ja	 Nein	 Ja	 Ja	 Ja
Amazon Message Delivery Service	 Ja	 Nein	 Nein	 Nein	 Ja	 Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Message Gateway-Service	Ja	Nein	Nein	Nein	Ja	Nein
AWS Microservice Extractor for .NET	Ja	Nein	Nein	Nein	Ja	Nein
AWS Punkte für das Migration Acceleration Program	Ja	Ja	Nein	Nein	Ja	Nein
AWS Migration Hub	Ja	Ja	Nein	Nein	Ja	Ja
AWS Migration Hub Orchestrator	Ja	Ja	Nein	Ja	Ja	Ja
AWS Migration Hub Refactor Spaces	Ja	Ja	Ja	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Migration Hub Strategy Recommendations	Ja	Nein	Nein	Nein	Ja	Ja
Amazon Monitron	Ja	Ja	Nein	Ja	Ja	Ja
Amazon MQ	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Neptune	Ja	Ja	Nein	Nein	Ja	Ja
Amazon Neptune Analytics	Ja	Ja	Nein	Ja	Ja	Nein
AWS Network Firewall	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Network Manager	Ja	Ja	Nein	Ja	Ja	Ja (Informationen)
AWS Network Manager Chatten	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Nimble Studio	Ja	Ja	Nein	Ja	Ja	Nein
Amazon One Enterprise	Ja	Ja	Nein	Ja	Ja	Nein
OpenSearchEinnahme durch Amazon	Ja	Ja	Nein	Ja	Ja	Ja
Amazon OpenSearch Serverlos	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
OpenSearch Amazon-Dienst	Ja	Ja	Ja	Ja	Ja	Ja
AWS OpsWorks	Ja	Ja	Nein	Nein	Ja	Nein
AWS OpsWorks Verwaltung der Konfiguration	Ja	Ja	Nein	Nein	Ja	Nein
AWS Organizations	Ja	Ja	Nein	Ja	Nein	Ja
AWS Outposts	Ja	Ja	Nein	Ja	Ja	Ja
AWS Panorama	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Partner Zentrale Kontoverwaltung	Ja	Nein	Nein	Nein	Ja	Nein
AWS Payment Cryptography	Ja	Ja	Nein	Ja	Ja	Nein
AWS Zahlungen	Ja	Nein	Nein	Nein	Ja	Nein
AWS Performance Insights	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Personalize	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Pinpoint	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Pinpoint-E-Mail-Service	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Pinpoint-SMS- und Sprachnachrichten-Service	Ja	Nein	Nein	Nein	Ja	Nein
Amazon-Pinpoint-SMS- und Sprachnachrichten-Service v2	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Polly	Ja	Ja	Nein	Nein	Ja	Nein
AWS-Preisliste	Ja	Nein	Nein	Nein	Ja	Nein
AWS Privates 5G	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Private CA Connector for Active Directory	Ja	Ja	Nein	Ja	Ja	Nein
AWS Private CA Anschluss für SCEP	Ja	Ja	Nein	Ja	Ja	Nein
AWS Private Certificate Authority (AWS Private CA)	Ja	Ja	Ja	Ja	Ja	Nein
AWS Proton	Ja	Ja	Nein	Ja	Ja	Ja
AWS Konsole für Bestellungen	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Q Business	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Q Business Q Apps	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Q-Entwickler	Ja	Nein	Nein	Nein	Ja	Ja
Amazon Q in Connect	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Quantum Ledger Database (Amazon QLDB)	Ja	Ja	Nein	Ja	Ja	Nein
Amazon QuickSight	Ja	Ja	Nein	Ja	Ja	Nein
Amazon RDS Daten-API	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon-RDS-IAM-Authentifizierung	Ja	Ja	Nein	Nein	Ja	Nein
AWS Papierkorb	Ja	Ja	Nein	Ja	Ja	Nein
Amazon-Redshift	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Redshift-Daten-API	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Redshift Serverless	Ja	Ja	Ja	Ja	Ja	Nein
Amazon Rekognition	Ja	Ja	Teilweise (Informationen)	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Relational Database Service (Amazon RDS) (Informationen)	Ja	Ja	Nein	Ja	Ja	Ja
AWS re:Post Privat	Ja	Ja	Nein	Ja	Ja	Ja
AWS Resilience Hub	Ja	Ja	Nein	Ja	Ja	Nein
AWS Resource Access Manager (AWS RAM)	Ja	Ja	Nein	Ja	Ja	Ja
AWS Ressourcen Explorer	Ja	Ja	Nein	Ja	Ja	Ja
AWS Resource Groups	Ja	Ja	Nein	Ja	Teilweise (Informationen)	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Resource Groups Tagging API	Ja	Nein	Nein	Nein	Ja	Nein
Amazon RHEL Knowledgebase Portal	Ja	Nein	Nein	Nein	Ja	Nein
AWS RoboMaker	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Route 53	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Route 53 Application Recovery Controller – Zonenwechseel	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Route 53-Domains	Ja	Nein	Nein	Nein	Nein	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Route 53-Profile	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Route 53 Recovery Cluster	Ja	Ja	Nein	Nein	Ja	Nein
Recovery-Kontroll-Config von Amazon Route 53	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Route 53 Recovery-Bereitschaft	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Route 53 Resolver	Ja	Ja	Nein	Ja	Ja	Ja
Amazon S3 Express	Ja	Ja	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon S3 Glacier	Ja	Ja	Ja	Ja	Ja	Nein
Amazon SageMaker	Ja	Ja	Nein	Ja	Ja	Teilweise (Informationen)
SageMaker Geospatial-Funktionen von Amazon	Ja	Ja	Nein	Ja	Ja	Nein
Amazon SageMaker Ground Truth Synthetisch	Ja	Nein	Nein	Nein	Ja	Nein
AWS Savings Plans	Ja	Ja	Nein	Ja	Ja	Nein
AWS Secrets Manager	Ja	Ja	Ja	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Security Hub	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Security Lake	Ja	Ja	Nein	Nein	Ja	Ja
AWS Security Token Service (AWS STS)	Ja	Teilweise (Informationen)	Nein	Ja	Teilweise (Informationen)	Nein
AWS Serverless Application Repository	Ja	Ja	Ja	Nein	Ja	Nein
AWS Service Catalog	Ja	Ja	Nein	Ja	Ja	Ja
Service Quotas	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Shield	Ja	Ja	Nein	Ja	Ja	Ja
AWS Signer	Ja	Ja	Ja	Ja	Ja	Nein
AWS Melden Sie sich an	Ja	Nein	Nein	Nein	Ja	Nein
Amazon SimpleDB	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Simple Email Service - Mail Manager	Ja	Ja	Nein	Ja	Ja	Ja
Amazon Simple Email Service (Amazon SES) v2	Ja	Teilweise (Informationen)	Ja	Ja	Teilweise (Informationen)	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon-Simple-Notification-Service (Amazon-SNS)	 Ja	 Ja	 Ja	 Ja	 Ja	 Nein
Amazon-Simple-Queue-Service (Amazon SQS)	 Ja	 Ja	 Ja	 Teilweise	 Ja	 Nein
Amazon-Simple-Storage-Service (Amazon-S3)	 Ja	 Ja	 Ja	 Teilweise (Informationen)	 Ja	 Teilweise (Informationen)
Amazon Simple Storage Service (Amazon S3) Object Lambda	 Ja	 Ja	 Nein	 Nein	 Ja	 Nein
Amazon Simple Storage Service (Amazon S3) auf AWS Outposts	 Ja	 Ja	 Ja	 Nein	 Ja	 Ja
Amazon Simple Workflow Service (Amazon SWF)	 Ja	 Ja	 Nein	 Ja	 Ja	 Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS SimSpaceWeber	Ja	Ja	Nein	Ja	Ja	Nein
AWS Site-to-Site VPN	Ja	Ja	Nein	Nein	Ja	Ja
AWS Snowball	Ja	Nein	Nein	Nein	Ja	Nein
AWS Snowball Edge	Ja	Nein	Nein	Nein	Ja	Nein
AWS Snow Device Management	Ja	Ja	Nein	Ja	Ja	Nein
AWS SQL Workbench	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Step Functions	Ja	Ja	Nein	Ja	Ja	Nein
AWS Storage Gateway	Ja	Ja	Nein	Ja	Ja	Nein
AWS Supply Chain	Ja	Ja	Nein	Ja	Ja	Nein
AWS Support App in Slack	Ja	Nein	Nein	Nein	Ja	Nein
AWS Support	Ja	Nein	Nein	Nein	Ja	Ja
AWS Support Pläne	Ja	Nein	Nein	Nein	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Support Empfehlungen	Ja	Nein	Nein	Nein	Ja	Nein
AWS Nachhaltigkeit	Ja	Nein	Nein	Nein	Ja	Nein
AWS Systems Manager	Ja	Ja	Nein	Ja	Ja	Ja
AWS Systems Manager für SAP	Ja	Ja	Nein	Ja	Ja	Nein
AWS Systems Manager GUI-Verbindung	Ja	Nein	Nein	Nein	Ja	Nein
AWS Systems Manager Incident Manager	Ja	Ja	Ja	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Systems Manager Incident Manager Kontakte	Ja	Ja	Ja	Nein	Ja	Nein
Tag Editor	Ja	Nein	Nein	Nein	Ja	Nein
AWS Steuereinstellungen	Ja	Nein	Nein	Nein	Ja	Nein
AWS Telco Network Builder	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Textract	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Timestream	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon Timestream Influxdb	Ja	Ja	Nein	Ja	Ja	Ja
AWS Tiros API (für Reachability Analyzer)	Ja	Nein	Nein	Nein	Nein	Nein
Amazon Transcribe	Ja	Ja	Nein	Ja	Ja	Nein
AWS Transfer Family	Ja	Ja	Nein	Ja	Ja	Nein
Amazon Translate	Ja	Ja	Nein	Ja	Ja	Nein
AWS Trusted Advisor	Teilweise (Informationen)	Ja	Nein	Nein	Teilweise	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Benachrichtigungen von Benutzern	Ja	Ja	Nein	Ja	Ja	Ja
AWS Benutzerbenachrichtigungen Kontakte	Ja	Ja	Nein	Ja	Ja	Nein
AWS Benutzerabonnements	Ja	Nein	Nein	Nein	Ja	Nein
AWS Verified Access	Ja	Nein	Nein	Nein	Ja	Nein
Amazon Verified Permissions	Ja	Ja	Nein	Nein	Ja	Nein
Amazon Virtual Private Cloud (Amazon VPC)	Ja	Teilweise (Informationen)	Teilweise (Informationen)	Ja	Ja	Teilweise (Informationen)

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon VPC Lattice	Ja	Ja	Nein	Ja	Ja	Nein
Amazon VPC Lattice Services	Ja	Ja	Nein	Nein	Ja	Nein
AWS WAF	Ja	Ja	Nein	Ja	Ja	Ja
AWS WAF Classic	Ja	Ja	Nein	Ja	Ja	Ja
AWS WAF Regional	Ja	Ja	Nein	Ja	Ja	Ja
AWS Well-Architected Tool	Ja	Ja	Nein	Ja	Ja	Nein

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
AWS Wickr	Ja	Ja	Nein	Ja	Ja	Nein
Amazon WorkDocs	Ja	Nein	Nein	Nein	Ja	Nein
Amazon WorkMail	Ja	Ja	Nein	Ja	Ja	Ja
WorkMail Amazon-Nachrichtenfluss	Ja	Ja	Nein	Nein	Ja	Nein
Amazon WorkSpaces	Ja	Ja	Nein	Ja	Ja	Nein
WorkSpaces Sicherer Browser von Amazon	Ja	Ja	Nein	Ja	Ja	Ja

Service	Aktionen	Berechtigungen auf Ressourcenebene	Ressourcenbasierte Richtlinien	ABAC	Temporäre Anmeldeinformationen	Serviceverknüpfte Rollen
Amazon WorkSpaces Thin Client	Ja	Ja	Nein	Ja	Ja	Nein
AWS X-Ray	Ja	Teilweise (Informationen)	Nein	Teilweise (Informationen)	Ja	Nein

Weitere Informationen

Amazon CloudFront

CloudFront hat keine dienstbezogenen Rollen, Lambda @Edge jedoch schon. Weitere Informationen finden Sie unter Service-Linked Roles for Lambda @Edge im Amazon CloudFront Developer Guide.

AWS CloudTrail

CloudTrail unterstützt ressourcenbasierte Richtlinien nur für CloudTrail Kanäle, die für CloudTrail Lake-Integrationen mit Ereignisquellen außerhalb von verwendet werden. AWS

CloudTrail unterstützt die Tag-basierte Zugriffskontrolle für CloudTrail Lake-Ereignisdatenspeicher und -Kanäle. CloudTrail unterstützt keine tagbasierten Zugriffskontrollen für Wanderwege.

Amazon CloudWatch

CloudWatch Rollen, die mit Services verknüpft sind AWS Management Console, können nicht mit der Funktion Alarmaktionen erstellt werden und unterstützen nur die Funktion Alarmaktionen.

AWS CodeBuild

CodeBuild unterstützt die kontenübergreifende gemeinsame Nutzung von Ressourcen mithilfe von. AWS RAM

CodeBuild unterstützt ABAC für projektbasierte Aktionen.

AWS Config

AWS Config unterstützt Berechtigungen auf Ressourcenebene für die Datenaggregation und Regeln für mehrere Konten in mehreren Regionen. AWS Config Eine Liste der unterstützten Ressourcen finden Sie im Bereich Datenaggregation über mehrere Konten und Regionen und im Bereich AWS Config -Regeln im AWS Config -API-Handbuch.

AWS Database Migration Service

Sie können Richtlinien erstellen und ändern, die mit AWS KMS Verschlüsselungsschlüsseln verknüpft sind, die Sie zur Verschlüsselung von Daten erstellen, die auf unterstützte Zielendpunkte migriert wurden. Die unterstützten Ziel-Endpunkte enthalten Amazon Redshift und Amazon S3. Weitere Informationen finden Sie unter Erstellen und Verwenden von AWS KMS Schlüsseln zur Verschlüsselung von Amazon Redshift Redshift-Zieldaten und Erstellen von AWS KMS Schlüsseln zur Verschlüsselung von Amazon S3 S3-Zielobjekten im AWS Database Migration Service Benutzerhandbuch.

Amazon Elastic Compute Cloud

Serviceverknüpfte Rollen von Amazon EC2 können nur für die folgenden Features verwendet werden: Spot-Instance-Anforderungen, Spot-Flotten-Anforderungen, Amazon-EC2-Flotten, und Schneller Start für Windows-Instances.

Amazon Elastic Container Service

Nur einige Amazon-ECS-Aktionen unterstützen Berechtigungen auf Ressourcenebene.

AWS Elemental MediaPackage

MediaPackage unterstützt dienstbezogene Rollen für die Veröffentlichung von Kundenzugriffsprotokollen, CloudWatch aber nicht für andere API-Aktionen.

AWS Identity and Access Management

IAM unterstützt nur eine Art ressourcenbasierte Richtlinie, die als Rollen-Vertrauensrichtlinie bezeichnet wird, die einer IAM-Rolle zugewiesen ist. Weitere Informationen finden Sie unter Erteilen von Berechtigungen an einen Benutzer zum Wechseln von Rollen.

IAM unterstützt die Tag-basierte Zugriffskontrolle nur für die meisten IAM-Ressourcen. Weitere Informationen finden Sie unter Markieren von IAM-Ressourcen.

Nur einige der API-Aktionen für IAM können mit temporären Anmeldeinformationen aufgerufen werden. Weitere Informationen finden Sie unter Vergleich Ihrer API-Optionen.

AWS IoT

Geräte, mit AWS IoT denen verbunden ist, werden mithilfe von X.509-Zertifikaten oder Amazon Cognito Identities authentifiziert. Sie können AWS IoT Richtlinien an ein X.509-Zertifikat oder Amazon Cognito Identity anhängen, um zu kontrollieren, wofür das Gerät autorisiert ist. Weitere Informationen finden Sie unter Sicherheits- und Identitätsmethoden für AWS IoT im AWS IoT Developer Guide

AWS Lambda

Lambda unterstützt attributbasierte Zugriffssteuerung (ABAC) für API-Aktionen, die eine Lambda-Funktion als erforderliche Ressource verwenden. Layers, Zuordnungen von Ereignisquellen und Konfigurationsressourcen für Codesignaturen werden nicht unterstützt.

Lambda verfügt im Gegensatz zu Lambda@Edge nicht über serviceverknüpfte Rollen. Weitere Informationen finden Sie unter Service-Linked Roles for Lambda @Edge im Amazon CloudFront Developer Guide.

Amazon Lightsail

Lightsail unterstützt teilweise Berechtigungen auf Ressourcenebene und ABAC. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Lightsail.

Amazon Managed Streaming for Apache Kafka (MSK)

Sie können eine Clusterrichtlinie an einen Amazon MSK-Cluster anhängen, der für Multi-VPC-Konnektivität konfiguriert wurde.

AWS Network Manager

AWS Cloud WAN unterstützt auch serviceverknüpfte Rollen. Weitere Informationen finden Sie unter Service-verknüpfte AWS Cloud-WAN-Rollen im Amazon VPC AWS Cloud WAN-Leitfaden.

Amazon Relational Database Service

Amazon Aurora ist eine vollständig verwaltete, mit MySQL und PostgreSQL kompatible relationale Datenbank-Engine. Sie können Aurora MySQL oder Aurora PostgreSQL als die DB-Engine-Option auswählen, wenn Sie über Amazon RDS neue Datenbankserver einrichten. Weitere Informationen finden Sie unter Identitäts- und Zugriffsverwaltung mit Amazon Aurora im Amazon-Aurora-Benutzerhandbuch.

Amazon Rekognition

Ressourcenbasierte Richtlinien werden nur für das Kopieren von Modellen von Amazon Rekognition Custom Labels unterstützt.

AWS Resource Groups

Benutzer können eine Rolle mit einer Richtlinie übernehmen, die Ressourcengruppen-Operationen zulässt.

Amazon SageMaker

Stellen im Zusammenhang mit Dienstleistungen sind derzeit für SageMaker Studio- und SageMaker Schulungsjobs verfügbar.

AWS Security Token Service

AWS STS verfügt nicht über „Ressourcen“, ermöglicht es jedoch, den Zugriff auf ähnliche Weise für Benutzer einzuschränken. Weitere Informationen finden Sie unter Verweigern des Zugriffs auf temporäre Sicherheitsanmeldeinformationen nach Name.

Nur einige API-Operationen AWS STS unterstützen das Aufrufen mit temporären Anmeldeinformationen. Weitere Informationen finden Sie unter Vergleich Ihrer API-Optionen.

Amazon Simple Email Service

Sie können nur Berechtigungen auf Ressourcenebene in Richtlinienanweisungen verwenden, die sich auf Aktionen beziehen, bei denen es um das Senden von E-Mails geht, beispielsweise

`ses:SendEmail` oder `ses:SendRawEmail`. Bei Richtlinienanweisungen, die sich auf andere Aktionen beziehen, kann das Ressourcenelement nur * enthalten.

Temporäre Anmeldeinformationen werden nur vom Amazon-SES-API unterstützt. Die Amazon SES SMTP-Schnittstelle unterstützt keine SMTP-Anmeldeinformationen, die von temporären Anmeldeinformationen stammen.

Amazon Simple Storage Service

Amazon S3 unterstützt die Tag-basierte Autorisierung nur für Objekt-Ressourcen.

Amazon S3 unterstützt serviceverknüpfte Rollen für Amazon S3 Storage Lens.

AWS Trusted Advisor

Der API-Zugriff auf Trusted Advisor erfolgt über die AWS Support API und wird durch AWS Support IAM-Richtlinien gesteuert.

Amazon Virtual Private Cloud

In einer IAM-Benutzerrichtlinie können Sie die Berechtigungen nicht auf einem bestimmten Amazon-VPC-Endpunkt einschränken. Für jedes `Action`-Element, das die API-Aktion `ec2:*VpcEndpoint*` oder `ec2:DescribePrefixLists` enthält, muss `""Resource": "*""` angegeben werden. Weitere Informationen finden Sie unter Identity and Access Management für VPC-Endpunkte und VPC-Endpunkt-Services im AWS PrivateLink -Leitfaden.

Amazon VPC unterstützt das Anfügen einer einzelnen Ressourcenrichtlinie zu einem VPC-Endpunkt, um den Zugriff über diesen Endpunkt einzuschränken. Weitere Informationen zur Verwendung ressourcenbasierter Richtlinien zum Steuern des Zugriffs auf Ressourcen von bestimmten Amazon-VPC-Endpunkten finden Sie unter Steuern des Zugriffs auf Services mithilfe von Endpunktrichtlinien im AWS PrivateLink -Handbuch.

Amazon VPC hat keine serviceverknüpften Rollen, tut es aber AWS Transit Gateway . Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für das Transit-Gateway im Amazon VPC-Leitfaden AWS Transit Gateway .

AWS X-Ray

X-Ray unterstützt nicht für alle Aktionen Berechtigungen auf Ressourcenebene.

X-Ray unterstützt Tag-basierte Zugriffskontrolle für Gruppen und Probenregeln.

AWS API-Anfragen signieren

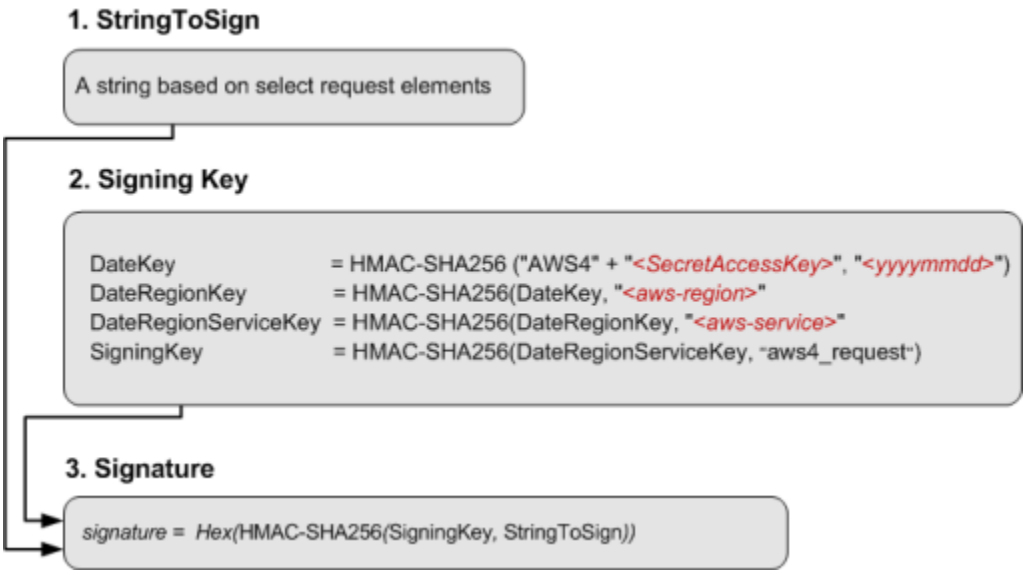
> **Important**
>
> Wenn Sie ein AWS SDKs (siehe Beispielcode und Bibliotheken) oder ein AWS Befehlszeilentool (CLI) zum Senden von API-Anfragen verwenden AWS, können Sie diesen Abschnitt überspringen, da das SDK und die CLI-Clients Ihre Anfragen mithilfe der von Ihnen bereitgestellten Zugriffsschlüssel authentifizieren. Sofern Sie keinen guten Grund dagegen haben, empfiehlt es sich, stets ein SDK oder die CLI zu verwenden.
> In Regionen, die mehrere Signaturversionen unterstützen, müssen Sie beim manuellen Signieren von Anfragen angeben, welche Signaturversion verwendet wird. Wenn Sie Anfragen an Multi-Region-Zugriffspunkte stellen, wechseln SDKs und die CLI ohne zusätzliche Konfiguration automatisch zur Verwendung der Signaturversion 4A.

Die Authentifizierungsinformationen, die Sie in einer Anfrage senden, müssen eine Signatur enthalten. Um eine Signatur zu berechnen, verketten Sie zunächst ausgewählte Anforderungselemente, um eine Zeichenfolge zu bilden, die als zu signierende Zeichenfolge bezeichnet wird. Anschließend verwenden Sie einen Signaturschlüssel, um den Hash-basierten Nachrichtenauthentifizierungscode (HMAC) der zu signierenden Zeichenfolge zu berechnen.

In AWS Signature Version 4 verwenden Sie Ihren geheimen Zugriffsschlüssel nicht, um die Anfrage zu signieren. Verwenden Sie stattdessen zunächst Ihren geheimen Zugriffsschlüssel, um einen Signierschlüssel abzuleiten. Der abgeleitete Signaturschlüssel ist spezifisch für Datum, Service und Region. Weitere Informationen zum Ableiten eines Signaturschlüssels in unterschiedlichen Programmiersprachen finden Sie unter Anfordern von Signaturbeispielen.

Signature Version 4 ist das AWS Signaturprotokoll. AWS unterstützt auch eine Erweiterung, Signature Version 4A, die Signaturen für API-Anfragen mit mehreren Regionen unterstützt. Weitere Informationen finden Sie im a-signing-examplesSigv4-Projekt unter. GitHub

Das folgende Diagramm veranschaulicht den allgemeinen Prozess der Berechnung einer Signatur.



- Die zu signierende Zeichenfolge hängt von der Art der Anfrage ab. Wenn Sie beispielsweise den HTTP-Autorisierungsheader oder die Abfrageparameter für die Authentifizierung verwenden, verwenden Sie eine variierende Kombination von Anforderungselementen, um die zu signierende Zeichenfolge zu erstellen. Bei einer HTTP-POST-Anfrage ist die `POST`-Richtlinie in der Anfrage die von Ihnen signierte Zeichenfolge.
- Für den Signaturschlüssel zeigt das Diagramm eine Reihe von Berechnungen, wobei das Ergebnis jedes Schritts in den nächsten Schritt einfließt. Der letzte Schritt ist der Signaturschlüssel.
- Wenn ein AWS Dienst eine authentifizierte Anfrage empfängt, erstellt er die Signatur anhand der in der Anfrage enthaltenen Authentifizierungsinformationen neu. Wenn die Signaturen übereinstimmen, verarbeitet der Service die Anforderung. Andernfalls wird die Anforderung abgelehnt.

Inhalt

Wann müssen Anforderungen signiert werden?

Wenn Sie benutzerdefinierten Code schreiben, an den API-Anfragen gesendet werden AWS, müssen Sie Code angeben, der die Anfragen signiert. Möglicherweise schreiben Sie benutzerdefinierten Code, weil:

- Sie arbeiten mit einer Programmiersprache, für die kein AWS SDK verfügbar ist.
- Sie benötigen die vollständige Kontrolle darüber, an welche Empfänger Anfragen gesendet werden AWS.

Warum werden Anforderungen signiert?

Das Signieren erhöht die Sicherheit der Anforderungen, und zwar auf folgende Weise:

- Überprüfung der Identität des Anforderers

  Für authentifizierte Anfragen ist eine Signatur erforderlich, die Sie mithilfe Ihrer Zugriffsschlüssel (Zugriffsschlüssel-ID, geheimer Zugriffsschlüssel) erstellen. Wenn Sie temporäre Sicherheitsanmeldeinformationen verwenden, ist für die Signaturberechnungen auch ein Sicherheitstoken erforderlich. Weitere Informationen finden Sie unter programmgesteuerter Zugriff auf AWS -Sicherheitsanmeldeinformationen.
- Schutz der Daten während der Übertragung

  Um zu verhindern, dass eine Anforderung während der Übertragung manipuliert wird, werden einige ihrer Elemente verwendet, um einen Hash-Wert (Digest) der Anforderung zu berechnen. Der daraus resultierende Hash-Wert wird in die Anforderung aufgenommen. Wenn an die Anfrage AWS-Service empfängt, verwendet es dieselben Informationen, um einen Hash zu berechnen, und vergleicht ihn mit dem Hashwert in Ihrer Anfrage. Wenn die Werte nicht übereinstimmen, wird die AWS Anfrage abgelehnt.
- Schutz vor potenziellen Replay-Angriffen

  In den meisten Fällen muss eine Anfrage AWS innerhalb von fünf Minuten nach dem Zeitstempel in der Anfrage eingehen. Andernfalls wird die AWS Anfrage abgelehnt.

Elemente einer AWS API-Anforderungssignatur

> **Important**
>
> Sofern Sie die AWS SDKs oder CLI nicht verwenden, müssen Sie Code schreiben, um Signaturen zu berechnen, die Authentifizierungsinformationen in Ihren Anfragen enthalten. Die Signaturberechnung in AWS Signature Version 4 kann ein komplexes Unterfangen sein, und wir empfehlen, dass Sie, wann immer möglich, die AWS SDKs oder CLI verwenden.

Jede HTTP/HTTPS-Anforderung, die Signature Version 4 verwendet, muss diese Elemente enthalten.

Elemente

- Endpunktspezifizierung
- Aktion
- Aktionsparameter
- Datum
- Authentifizierungsinformationen

Endpunktspezifizierung

Gibt den DNS-Namen des Endpunkts an, an den Sie die Anforderung senden. Dieser Name enthält normalerweise den Servicecode und die Region. Der Endpunkt für Amazon DynamoDB in der `us-east-1`-Region ist beispielsweise `dynamodb.us-east-1.amazonaws.com`.

Für HTTP/1.1-Anforderungen müssen Sie den `Host`-Header einfügen. Für HTTP/2-Anforderungen können Sie den `:authority`-Header oder den `Host`-Header einfügen. Wenn die Anforderung der HTTP/2-Spezifikation entsprechen soll, verwenden Sie nur den `:authority`-Header. Nicht alle Services unterstützen HTTP/2-Anforderungen.

Informationen zu den Endpunkten, die von den einzelnen Services unterstützt werden, finden Sie unter Service-Endpunkte und Kontingente in der Allgemeine AWS-Referenz.

Aktion

Gibt eine API-Aktion für den Service an. Zum Beispiel die DynamoDB `CreateTable`-Aktion oder die Amazon-EC2 `DescribeInstances`-Aktion.

Informationen zu den Aktionen, die von den einzelnen Services unterstützt werden, finden Sie in der Service-Autorisierungsreferenz.

Aktionsparameter

Gibt die Parameter für die in der Anforderung angegebene Aktion an. Jede AWS API-Aktion hat eine Reihe erforderlicher und optionaler Parameter. Die API-Version ist in der Regel ein erforderlicher Parameter.

Informationen zu den von einer API-Aktion unterstützten Parametern finden Sie in der API-Referenz für den Service.

Datum

Gibt das Datum und die Uhrzeit der Anforderung an. Indem Sie der Anforderung Uhrzeit und Datum hinzufügen, können Sie verhindern, das Dritte Ihre Anforderung abfangen und zu einem späteren Zeitpunkt erneut hochladen. Das Datum, das Sie im Umfang der Anmeldeinformationen angeben, muss mit dem Datum Ihrer Anforderung übereinstimmen.

Der Zeitstempel muss in UTC angegeben werden und das folgende ISO-8601-Format verwenden: JJJJMMDDTHHMMSSZ. z. B. `20220830T123600Z`. Geben Sie im Zeitstempel keine Millisekunden an.

Sie können ein `date` oder einen `x-amz-date`-Header verwenden oder `x-amz-date` als Abfrageparameter einfügen. Wenn wir keinen `x-amz-date`-Header finden können, suchen wir nach einem `date`-Header.

Authentifizierungsinformationen

Jede Anfrage, die Sie senden, muss die folgenden Informationen enthalten. AWS verwendet diese Informationen, um die Gültigkeit und Authentizität der Anfrage sicherzustellen.

- Algorithmus – Verwenden Sie `AWS 4-HMAC-SHA256`, um Signature Version 4 mit dem `HMAC-SHA256`-Hash-Algorithmus anzugeben.
- Anmeldeinformationen – Eine Zeichenfolge, die aus Ihrer Zugriffsschlüssel-ID, dem Datum im Format JJJJMMTT, dem Regionscode, dem Servicecode und der durch Schrägstriche (/) getrennte `aws4_request`-Abschlusszeichenfolge besteht. Für den Regionscode, den Servicecode und die Abschlusszeichenfolge müssen Kleinbuchstaben verwendet werden.

```
AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
```

- Signierte Header – Die HTTP-Header, die in die Signatur aufgenommen werden sollen, getrennt durch Semikolons (;). z. B. `host;x-amz-date`.
- Signatur – Eine hexadezimalkodierte Zeichenfolge, die die berechnete Signatur darstellt. Sie müssen die Signatur mit dem Algorithmus berechnen, den Sie im `Algorithm`-Parameter angegeben haben.

Authentifizierungsmethoden

> **Important**
>
> Sofern Sie die AWS SDKs oder CLI nicht verwenden, müssen Sie Code schreiben, um Signaturen zu berechnen, die Authentifizierungsinformationen in Ihren Anfragen enthalten. Die Signaturberechnung in AWS Signature Version 4 kann ein komplexes Unterfangen sein, und wir empfehlen, dass Sie, wann immer möglich, die AWS SDKs oder CLI verwenden.

Mit einer der folgenden Methoden können Sie Authentifizierungsinformationen ausdrücken.

HTTP-Autorisierungs-Header

Der HTTP-`Authorization`-Header ist die gängigste Methode zur Authentifizierung einer Anfrage. Alle REST-API-Vorgänge (außer browserbasierte Uploads mit `POST`-Anfragen) erfordern diesen Header. Weitere Informationen zum Autorisierungsheader-Wert und zur Berechnung der Signatur und verwandter Optionen finden Sie unter Authentifizieren von Anfragen: Verwenden des Autorisierungsheaders (AWS Signature Version 4) in der Amazon S3 S3-API-Referenz.

Nachfolgend finden Sie ein Beispiel für den `Authorization`-Header-Wert. Zur besseren Lesbarkeit werden diesem Beispiel Zeilenumbrüche hinzugefügt. In Ihrem Code muss der Header eine fortlaufende Zeichenfolge sein. Zwischen dem Algorithmus und den Anmeldeinformationen steht kein Komma, die anderen Elemente müssen jedoch durch Kommas getrennt werden.

```
Authorization: AWS 4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Die folgende Tabelle beschreibt die verschiedenen Komponenten des Werts des Autorisierungs-Headers aus dem vorangegangenen Beispiel:

Komponente	Beschreibung
Autorisierung	Der Algorithmus, der zur Berechnung der Signatur verwendet wurde. Sie müssen diesen Wert angeben, wenn Sie AWS Signature Version 4 für die Authentifizierung verwenden . Die Zeichenfolge gibt AWS Signature Version 4 (`AWS 4`) und den Signaturalgorithmus () an. `HMAC-SHA256`
Credential	Ihre Zugriffsschlüssel-ID und die Informati onen des Gültigkeitsbereichs, darunter Datum, Region und Service, die zur Berechnung der Signatur verwendet wurden. Diese Zeichenfolge verfügt über das folgende Format: `<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request` Dabei gilt: Der `<date>`-Wert wird im Format JJJJMMTT angegeben. Der `<aws-serv ice>` -Wert ist s3, wenn eine Anfrage an Amazon S3 gesendet wird.
SignedHeaders	Eine durch Semikolons getrennte Liste von Anforderungs-Headern, die Sie zur Berechnun g von `Signature` verwendet haben. Die Liste enthält nur Header-Namen und die Header-Na men müssen in Kleinbuchstaben geschrieben sein. Zum Beispiel: host; range; x-amz-date
Signatur	Die 256-Bit-Signatur, ausgedrückt als 64 hexadezimale Kleinbuchstaben. Beispiel: `fe5f80f77d5fa3beca038a248ff`

Komponente	Beschreibung
	`027d0445342fe2855ddc963176630326f1024` Beachten Sie, dass die Signaturberechnungen je nach gewählter Option zur Übertragung der Nutzlast variieren.

Parameter der Abfragezeichenfolge

Sie können eine Abfragezeichenfolge verwenden, um eine Anfrage vollständig in einer URL auszudrücken. In diesem Fall verwenden Sie Abfrageparameter, um Anforderungsinformationen bereitzustellen, einschließlich der Authentifizierungsinformationen. Da die Anforderungssignatur Teil der URL ist, wird diese Art von URL oft als vorsignierte URL bezeichnet. Sie können vorsignierte URLs verwenden, um anklickbare Links in HTML einzubetten, die bis zu sieben Tage gültig sein können. Weitere Informationen finden Sie unter Authentifizieren von Anfragen: Verwenden von Abfrageparametern (AWS Signature Version 4) in der Amazon S3 S3-API-Referenz.

Im Folgenden finden Sie ein Beispiel für eine vorsignierte URL. Zur besseren Lesbarkeit werden diesem Beispiel Zeilenumbrüche hinzugefügt:

```
https://s3.amazonaws.com/examplebucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

> **Note**
>
> Der `X-Amz-Credential`-Wert in der URL zeigt das Zeichen „/“ nur zur besseren Lesbarkeit an. In der Praxis sollte es als %2F codiert werden. Beispielsweise: `&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request`

Die folgende Tabelle beschreibt die Abfrageparameter in der URL, die Informationen zur Authentifizierung bereitstellen.

Abfragezeichenfolgen-Parametername	Beschreibung
X-Amz-Algorithm	Identifiziert die Version von AWS Signature und den Algorithmus, den Sie zur Berechnung der Signatur verwendet haben. Für AWS Signature Version 4 setzen Sie diesen Parameterwert auf. `AWS 4-HMAC-SHA256` Diese Zeichenfolge identifiziert AWS -Signature Version 4 (AWS 4) und den HMAC-SHA256-Algorithmus (`HMAC-SHA256`).
X-Amz-Credential	Neben Ihrer Zugriffsschlüssel-ID gibt dieser Parameter auch den Bereich (AWS Region und Dienst) an, für den die Signatur gültig ist. Dieser Wert muss mit dem Gültigkeitsbereich übereinstimmen, den Sie für die Signaturb erechnungen verwenden, die im folgenden Abschnitt beschrieben werden. Das allgemeine Format für diesen Parameter wert lautet wie folgt: `<your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request` Beispiel: `AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request` Eine Liste der AWS regionalen Zeichenketten finden Sie unter Regionale Endpunkte in der AWS Allgemeinen Referenz.

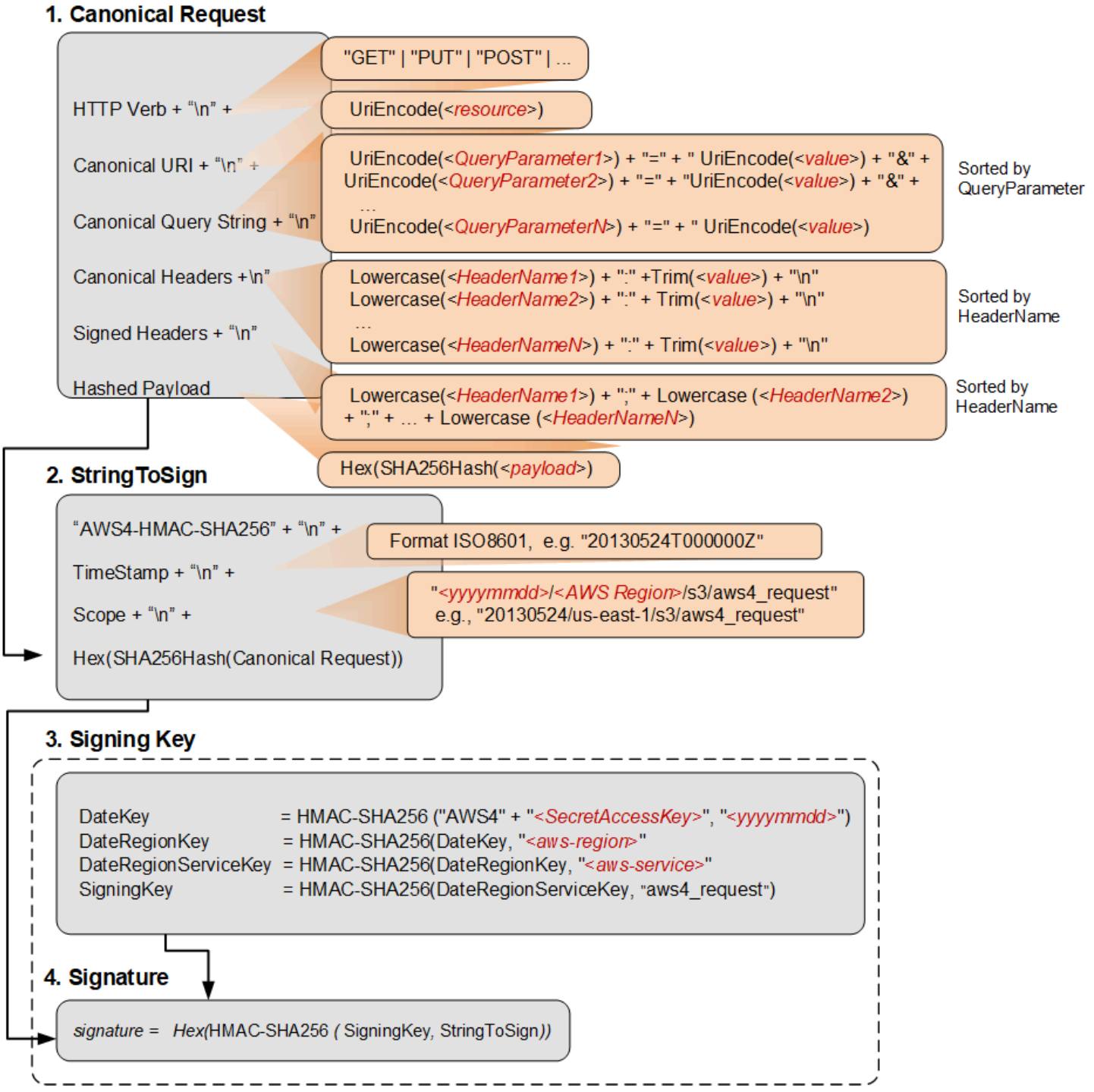
Abfragezeichenfolgen-Parametername	Beschreibung
X-Amz-Datum	Das Datums- und Uhrzeitformat muss dem ISO 8601-Standard entsprechen und im `yyyyMMddTHHmmssZ` -Format formatiert sein. Wenn Datum und Uhrzeit beispielsweise „01.08.2016 15:32:41.982-700“ lauten, müssen diese zunächst in UTC (koordinierte Weltzeit) konvertiert und dann als „20160801T223241Z“ übermittelt werden.
X-Amz-Expires	Gibt den Zeitraum in Sekunden an, für den die generierte vorsignierte URL gültig ist. Zum Beispiel 86 400 (24 Stunden). Es handelt sich um einen Ganzzahlwert. Der Mindestwert, den Sie festlegen können, ist 1 und der Höchstwer t 604 800 (sieben Tage). Eine vorsignierte URL kann maximal sieben Tage gültig sein, da der Signaturschlüssel, den Sie bei der Signaturb erechnung verwenden, bis zu sieben Tage gültig ist.
X-Amz- SignedHeaders	Listet die Header auf, die Sie zur Berechnun g der Signatur verwendet haben. Für die Signaturberechnungen sind folgende Header erforderlich: • Der HTTP-Host-Header. • Alle x-amz-*-Header, die Sie der Anfrage hinzufügen möchten. Für zusätzliche Sicherheit sollten Sie alle Anforderungs-Header signieren, die Sie in Ihre Anfrage aufnehmen möchten.

Abfragezeichenfolgen-Parametername	Beschreibung
X-Amz-Signature	Stellt die Signatur zur Authentifizierung Ihrer Anfrage bereit. Diese Signatur muss mit der vom Service berechneten Signatur übereinst immen. Andernfalls lehnt der Service die Anfrage ab. Beispiel: `733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7` Signaturberechnungen werden im folgenden Abschnitt beschrieben.
X-Amz-Security-Token	Optionaler Parameter für Anmeldeinformation en, wenn Sie Anmeldeinformationen verwenden, die vom STS-Service stammen.

Erstellen Sie eine signierte AWS API-Anfrage

> **Important**
>
> Wenn Sie ein AWS SDKs (siehe Beispielcode und Bibliotheken) oder ein AWS Befehlszeilentool (CLI) zum Senden von API-Anfragen verwenden AWS, können Sie diesen Abschnitt überspringen, da das SDK und die CLI-Clients Ihre Anfragen mithilfe der von Ihnen bereitgestellten Zugriffsschlüssel authentifizieren. Sofern Sie keinen guten Grund dagegen haben, empfiehlt es sich, stets ein SDK oder die CLI zu verwenden.
> In Regionen, die mehrere Signaturversionen unterstützen, müssen Sie beim manuellen Signieren von Anfragen angeben, welche Signaturversion verwendet wird. Wenn Sie Anfragen an Multi-Region-Zugriffspunkte stellen, wechseln SDKs und die CLI ohne zusätzliche Konfiguration automatisch zur Verwendung der Signaturversion 4A.

Im Folgenden finden Sie eine Übersicht über den Prozess zur Erstellung einer signierten Anforderung. Um eine Signatur zu berechnen, benötigen Sie zunächst eine zu signierende Zeichenfolge. Anschließend berechnen Sie mithilfe eines Signaturschlüssels einen HMAC-SHA256-Hash der zu signierenden Zeichenfolge. Das folgende Diagramm veranschaulicht den Prozess, einschließlich der verschiedenen Komponenten der Zeichenfolge, die Sie zum Signieren erstellen.



In der folgenden Tabelle werden die im Diagramm dargestellten Funktionen beschrieben. Für diese Funktionen muss Code implementiert werden. Weitere Informationen finden Sie in den Codebeispielen in den AWS SDKs.

Funktion	Beschreibung
`Lowercase()`	Wandeln Sie die Zeichenfolge in Kleinbuch staben um.
`Hex()`	Kodierung in Kleinbuchstaben im Basis-16-Format.
`SHA256Hash()`	Kryptografische Hash-Funktion des Secure Hash Algorithm (SHA).
`HMAC-SHA256()`	Berechnet HMAC unter Verwendung des SHA256-Algorithmus mit dem bereitgestellten Signaturschlüssel. Dies ist die endgültige Signatur.
`Trim()`	Entfernen Sie alle führenden oder nachgeste llten Leerzeichen.
`UriEncode()`	URI kodieren jedes Byte. UriEncode() muss die folgenden Regeln durchsetzen: • URI kodiert jedes Byte mit Ausnahme der nicht reservierten Zeichen: „A“–„Z“, „a“–„z“, „0“–„9“, „-“, „.“, „_“ und „~“. • Das Leerzeichen ist ein reserviertes Zeichen und muss als „%20“ (und nicht als „+“) codiert werden. • Jedes URI-codierte Byte besteht aus einem „%“ und dem zweistelligen Hexadezimalwert des Bytes. • Buchstaben im Hexadezimalwert müssen in Großbuchstaben geschrieben sein, zum Beispiel „%1A“. • Kodieren Sie den Schrägstrich „/“ überall außer im Objektschlüsselnamen. Wenn der Objektschlüsselname beispiels

Funktion	Beschreibung
	weise `photos/Jan/sample.jpg` lautet, wird der Schrägstrich im Schlüsselnamen nicht codiert. **Important** Die von Ihrer Entwicklungsplattform bereitgestellten UriEncode Standardf unktionen funktionieren möglicher weise aufgrund von Unterschieden in der Implementierung und der damit verbundenen Unklarheit in den zugrunde liegenden RFCs nicht. Wir empfehlen Ihnen, Ihre eigene benutzerdefinierte UriEncode Funktion zu schreiben, um sicherzustellen, dass Ihre Kodierung funktioniert. Ein Beispiel für eine UriEncode Funktion in Java finden Sie unter Java Utilities auf der GitHub Website.

Note

Beim Signieren Ihrer Anfragen können Sie entweder AWS Signature Version 4 oder AWS Signature Version 4A verwenden. Der Hauptunterschied zwischen den beiden wird dadurch bestimmt, wie die Signatur berechnet wird. Bei AWS Signature Version 4A enthält die Signatur keine regionsspezifischen Informationen und wird mithilfe des `AWS 4-ECDSA-P256-SHA256` Algorithmus berechnet.

Temporäre Sicherheitsanmeldeinformationen

Anstatt langfristige Anmeldeinformationen zum Signieren einer Anfrage zu verwenden, können Sie temporäre Sicherheitsanmeldeinformationen verwenden, die von AWS Security Token Service ()AWS STS bereitgestellt werden.

Wenn Sie temporäre Sicherheitsanmeldeinformationen verwenden, müssen Sie dem Autorisierungsheader oder der Abfragezeichenfolge das `X-Amz-Security-Token` hinzufügen, um das Sitzungstoken aufzunehmen. Bei einigen Services müssen Sie die kanonische Anforderung durch das `X-Amz-Security-Token` ergänzen. Bei anderen Services müssen Sie nur am Ende, nach der Berechnung der Signatur, `X-Amz-Security-Token` hinzufügen. Einzelheiten finden Sie in der jeweiligen AWS-Service Dokumentation.

Zusammenfassung der Signierschritte

Schritt 1: Erstellen einer kanonischen Anforderung

Ordnen Sie den Inhalt Ihrer Anfrage (Host, Aktion, Header usw.) in einem standardmäßigen kanonischen Format an. Die kanonische Anforderung ist einer der Eingabewerte, anhand derer eine zu signierende Zeichenfolge erstellt wird. Details hierzu finden Sie unter Elemente einer AWS API-Anforderungssignatur.

Schritt 2: Erstellen eines Hashes der kanonischen Anforderung

Leiten Sie einen Signaturschlüssel ab, indem Sie eine Reihe von verschlüsselten Hash-Operationen (HMAC-Operationen) am Anforderungsdatum, in der Region und im Dienst ausführen, wobei Ihr AWS geheimer Zugriffsschlüssel der Schlüssel für den ersten Hashing-Vorgang ist.

Schritt 3: Erstellen einer Zeichenfolge zum Signieren

Erstellen Sie eine zu signierende Zeichenfolge mit der kanonischen Anforderung und zusätzlichen Informationen wie Algorithmus, Anforderungsdatum, Umfang der Anmeldeinformationen und Digest (Hash) der kanonischen Anforderung.

Schritt 4: Berechnen der Signatur

Nachdem Sie den Signaturschlüssel abgeleitet haben, berechnen Sie die Signatur, indem Sie eine Keyed-Hash-Operation für die zu signierende Zeichenfolge durchführen. Verwenden Sie den abgeleiteten Signaturschlüssel als Hash-Schlüssel für diese Operation.

Schritt 5: Hinzufügen der Signatur zur Anforderung

Nachdem Sie die Signatur berechnet haben, fügen Sie sie einem HTTP-Header oder der Abfragezeichenfolge der Anforderung hinzu.

Schritt 1: Erstellen einer kanonischen Anforderung

Erstellen Sie eine kanonische Anforderung, indem Sie die folgenden Zeichenketten, getrennt durch Zeilenumbrüche, verketten. Dadurch wird sichergestellt, dass die von Ihnen berechnete Signatur und die berechnete Signatur übereinstimmen können. AWS

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod* – Die HTTP-Methode, z. B. `GET`, `PUT`, `HEAD` und `DELETE`.
- *CanonicalUri*— Die URI-kodierte Version des absoluten Pfadkomponenten-URI, beginnend mit dem „/“, das auf den Domainnamen folgt, bis zum Ende der Zeichenfolge oder bis zum Fragezeichen ('?') wenn Sie Parameter für die Abfragezeichenfolge haben. Wenn der absolute Pfad leer ist, verwenden Sie einen umgekehrten Schrägstrich (/). Beim URI im folgenden Beispiel, `/examplebucket/myphoto.jpg`, handelt es sich um den absoluten Pfad und Sie codieren das "/" nicht im absoluten Pfad:

  ```
  http://s3.amazonaws.com/examplebucket/myphoto.jpg
  ```

- *CanonicalQueryString*— Die URI-codierten Abfragezeichenfolgenparameter. Sie kodieren jeden Namen und jeden Wert einzeln mit dem URI. Sie müssen die Parameter in der kanonischen Abfragezeichenfolge außerdem alphabetisch nach Schlüsselnamen sortieren. Die Sortierung erfolgt nach der Codierung. Die Abfragezeichenfolge im folgenden URI-Beispiel lautet:

  ```
  http://s3.amazonaws.com/examplebucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  Die kanonische Abfragezeichenfolge lautet wie folgt (zur besseren Lesbarkeit wurden diesem Beispiel Zeilenumbrüche hinzugefügt):

  ```
  UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
  ```

```
UriEncode("max-keys")+"="+UriEncode("20") + "&" +
UriEncode("prefix")+"="+UriEncode("somePrefix")
```

Wenn eine Anfrage auf eine Unterressource abzielt, ist der entsprechende Abfrageparameterwert eine leere Zeichenfolge („“). Der folgende URI identifiziert beispielsweise die ACL-Unterressource im `examplebucket`-Bucket:

```
http://s3.amazonaws.com/examplebucket?acl
```

Das ist CanonicalQueryString in diesem Fall wie folgt:

```
UriEncode("acl") + "=" + ""
```

Wenn der URI kein „?“ enthält, gibt es in der Anfrage keine Abfragezeichenfolge und Sie legen die kanonische Abfragezeichenfolge auf eine leere Zeichenfolge („“) fest. Sie müssen weiterhin das „\n“ einfügen.

- *CanonicalHeaders*— Eine Liste von Anforderungsheadern mit ihren Werten. Einzelne Header-Namens- und Wertpaare werden durch das Zeilenumbruchzeichen („\n“) getrennt. Im Folgenden finden Sie ein Beispiel für einen Canonicalheader:

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

CanonicalHeaders Die Liste muss Folgendes enthalten:

- HTTP-`host`-Header.
- Wenn der `Content-Type` Header in der Anfrage vorhanden ist, müssen Sie ihn der *CanonicalHeaders*Liste hinzufügen.
- Alle `x-amz-*`-Header, die Sie in Ihre Anfrage aufnehmen möchten, müssen ebenfalls hinzugefügt werden. Wenn Sie beispielsweise temporäre Sicherheitsanmeldeinformationen verwenden, müssen Sie `x-amz-security-token` in Ihre Anfrage einschließen. Sie müssen diesen Header zur Liste von hinzufügen *CanonicalHeaders*.

> **Note**
>
> Der `x-amz-content-sha256` Header ist für Amazon S3 AWS S3-Anfragen erforderlich. Es stellt einen Hash der Anforderungsnutzlast bereit. Wenn keine Nutzdaten vorhanden sind, müssen Sie den Hash einer leeren Zeichenfolge bereitstellen.

Jeder Header-Name muss:

- Kleinbuchstaben verwenden.
- in alphabetischer Reihenfolge erscheinen.
- von einem Doppelpunkt (:) gefolgt sein.

Bei Werten müssen Sie:

- alle führenden oder nachgestellten Leerzeichen entfernen.
- aufeinanderfolgende Leerzeichen in ein einzelnes Leerzeichen umwandeln.
- die Werte für einen mehrwertigen Header durch Kommas trennen.
- Sie müssen den Host-Header (HTTP/1.1) oder den :authority-Header (HTTP/2) sowie alle `x-amz-*`-Header in die Signatur einschließen. Sie können optional andere Standard-Header in die Signatur aufnehmen, z. B. content-type.

Die in diesem Beispiel verwendeten `Lowercase()`- und `Trim()`-Funktionen werden im vorangehenden Abschnitt beschrieben.

Im Folgenden finden Sie ein Beispiel für eine `CanonicalHeaders`-Zeichenfolge. Die Header-Namen sind in Kleinbuchstaben geschrieben und sortiert.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Für die Berechnung einer Autorisierungssignatur sind nur der Host und etwaige `x-amz-*`-Header erforderlich. Um jedoch Datenmanipulationen vorzubeugen, sollten Sie erwägen, alle Header in die Signaturberechnung einzubeziehen.

- *`SignedHeaders`*— Eine alphabetisch sortierte, durch Semikolons getrennte Liste von Anforderungs-Header-Namen in Kleinbuchstaben. Bei den Anforderungs-Headern in der Liste handelt es sich um dieselben Header, die Sie in die Zeichenfolge `CanonicalHeaders` eingeschlossen haben. Für das vorherige Beispiel wäre der Wert von beispielsweise wie folgt: *`SignedHeaders`*

```
host;x-amz-content-sha256;x-amz-date
```

- *`HashedPayload`*— Eine Zeichenfolge, die unter Verwendung der Nutzdaten im Hauptteil der HTTP-Anfrage als Eingabe für eine Hash-Funktion erstellt wurde. Diese Zeichenfolge verwendet Hexadezimalzeichen in Kleinbuchstaben.

```
Hex(SHA256Hash(<payload>)
```

  Wenn die Anfrage keine Nutzlast enthält, berechnen Sie einen Hash der leeren Zeichenfolge wie folgt:

```
Hex(SHA256Hash(""))
```

  Der Hash gibt den folgenden Wert zurück:

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

  Wenn Sie beispielsweise ein Objekt mithilfe einer PUT-Anfrage hochladen, stellen Sie Objektdaten im Textkörper bereit. Wenn Sie ein Objekt mithilfe einer `GET`-Anfrage abrufen, berechnen Sie den Hash der leeren Zeichenfolge.

Schritt 2: Erstellen eines Hashes der kanonischen Anforderung

Erstellen Sie einen Hash (Digest) der kanonischen Anforderung mithilfe desselben Algorithmus, den Sie zur Erstellung des Hashes der Nutzdaten verwendet haben. Der Hash der kanonischen Anfrage ist eine Zeichenfolge aus hexadezimalen Zeichen in Kleinbuchstaben.

Schritt 3: Erstellen einer zu signierenden Zeichenfolge

Erstellen Sie eine Zeichenfolge, indem Sie die folgenden Zeichenfolgen, getrennt durch Zeilenumbrüche, verketten. Beenden Sie diese Zeichenfolge nicht mit einem Zeilenumbruch.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *Algorithmus* – Der Algorithmus, der verwendet wird, um den Hash der kanonischen Anforderung zu erstellen. Für SHA-256 ist der Algorithmus `AWS4-HMAC-SHA256`.
- *RequestDateTime*— Das Datum und die Uhrzeit, die im Bereich der Anmeldeinformationen verwendet wurden. Bei diesem Wert handelt es sich um die aktuelle UTC-Zeit im ISO 8601-Format (zum Beispiel `20130524T000000Z`).
- *CredentialScope*— Der Gültigkeitsbereich der Anmeldeinformationen. Dies schränkt die resultierende Signatur auf die angegebene Region und den angegebenen Service ein. Die Zeichenfolge verfügt über das folgende Format: *JJJJMMDD*/*region*/*service*/aws4_request.
- *HashedCanonicalRequest*— Der Hash der kanonischen Anfrage. Dieser Wert wird in Schritt 2 berechnet.

Im Folgenden finden Sie ein Beispiel für eine zu signierende Zeichenfolge.

```
"AWS4-HMAC-SHA256" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(SHA256Hash(<CanonicalRequest>))
```

Schritt 4: Berechnen der Signatur

In AWS Signature Version 4 verwenden Sie nicht Ihre AWS Zugangsschlüssel zum Signieren einer Anfrage, sondern erstellen als Authentifizierungsinformationen, die Sie Ihrer Anfrage hinzufügen, einen Signaturschlüssel, der auf eine bestimmte Region und einen bestimmten Dienst beschränkt ist.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Eine Liste der Regionszeichenfolgen finden Sie unter Regionale Endpunkte in der Allgemeinen AWS -Referenz.

Rufen Sie für jeden Schritt die Hash-Funktion mit den erforderlichen Schlüsseln und Daten auf. Das Ergebnis jedes Aufrufs der Hash-Funktion wird zur Eingabe für den nächsten Aufruf der Hash-Funktion.

Erforderliche Eingabe

- Eine Zeichenfolge (`Key`), die Ihren geheimen Zugriffsschlüssel enthält
- Eine Zeichenfolge (`Date`), die das im Gültigkeitsbereich der Anmeldeinformationen verwendete Datum im Format JJJJMMDD enthält
- Eine Zeichenfolge (`Region`), die den Regionscode enthält (z. B. `us-east-1`)
- Eine Zeichenfolge (`Service`), die den Servicecode enthält (z. B. `ec2`)
- Die zu signierende Zeichenfolge, die Sie im vorherigen Schritt erstellt haben.

So berechnen Sie die Signatur

1. Verketten Sie „AWS4" und den geheimen Zugriffsschlüssel. Rufen Sie die Hash-Funktion mit der verketteten Zeichenfolge als Schlüssel und der Datumszeichenfolge als Daten auf.

   ```
   kDate = hash("AWS4" + Key, Date)
   ```

2. Rufen Sie die Hash-Funktion mit dem Ergebnis des vorherigen Aufrufs als Schlüssel und der Region-Zeichenfolge als Daten auf.

   ```
   kRegion = hash(kDate, Region)
   ```

3. Rufen Sie die Hash-Funktion mit dem Ergebnis des vorherigen Aufrufs als Schlüssel und der Service-Zeichenfolge als Daten auf.

```
kService = hash(kRegion, Service)
```

4. Rufen Sie die Hash-Funktion mit dem Ergebnis des vorherigen Aufrufs als Schlüssel und „aws4_request“ als Daten auf.

```
kSigning = hash(kService, "aws4_request")
```

5. Rufen Sie die Hash-Funktion mit dem Ergebnis des vorherigen Aufrufs als Schlüssel und der zu signierenden Zeichenfolge als Daten auf. Das Ergebnis ist die Signatur als Binärwert.

```
signature = hash(kSigning, string-to-sign)
```

6. Konvertieren Sie die Signatur von der binären in die hexadezimale Darstellung in Kleinbuchstaben.

Schritt 5: Hinzufügen der Signatur zur Anforderung

Example Beispiel: Autorisierungsheader

Das folgende Beispiel zeigt einen `Authorization`-Header für die `DescribeInstances`-Aktion. Aus Gründen der Lesbarkeit ist dieses Beispiel mit Zeilenumbrüchen formatiert. In Ihrem Code muss dies eine fortlaufende Zeichenfolge sein. Es steht kein Komma zwischen Algorithmus und `Credential`. Die anderen Elemente müssen jedoch durch Kommas getrennt werden.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

Example Beispiel: Anforderung mit Authentifizierungsparametern in der Abfragezeichenfolge

Das folgende Beispiel zeigt eine Abfrage für die `DescribeInstances`-Aktion, die die Authentifizierungsinformationen enthält. Aus Gründen der Lesbarkeit ist dieses Beispiel mit Zeilenumbrüchen formatiert und ist nicht URL-codiert. In Ihrem Code muss die Abfragezeichenfolge eine fortlaufende Zeichenfolge sein, die URL-codiert ist.

```
https://ec2.amazonaws.com/?
```

```
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

Quellcode in den SDKs AWS

Die AWS SDKs enthalten Quellcode GitHub zum Signieren von AWS API-Anfragen. Codebeispiele finden Sie unter Beispielprojekte im AWS Beispiel-Repository.

- AWS SDK for .NET — AWS4Signer.cs
- AWS SDK for C++ — AWSAuthV4Signer.cpp
- AWS SDK for Go — v4.go
- AWS SDK for Java — 4Signer.java BaseAws
- AWS SDK for JavaScript — v4.js
- AWS SDK for PHP — SignatureV4.php
- AWS SDK for Python (Boto) — signers.py
- AWS SDK for Ruby — signer.rb

Anfordern von Signaturbeispielen

Die folgenden Beispiele für AWS Signieranfragen zeigen Ihnen, wie Sie SigV4 verwenden können, um Anfragen zu signieren, die ohne das AWS SDK oder das AWS Befehlszeilentool gesendet wurden.

Browserbasierter Amazon-S3-Upload mit HTTP POST

Unter Authenticating Requests: Browser-Based Uploads werden die Signatur und die relevanten Informationen beschrieben, die Amazon S3 verwendet, um die Signatur nach Erhalt der Anforderung zu berechnen.

Beispiel: Der browserbasierte Upload mit HTTP POST (unter Verwendung von AWS Signature Version 4) bietet weitere Informationen mit einer Beispiel-POST-Richtlinie und einem Formular, das Sie zum Hochladen einer Datei verwenden können. Die Beispielrichtlinie und die fiktive

Anmeldeinformation zeigen Ihnen den Workflow und die daraus resultierende Signatur und den Richtlinien-Hash.

Authentifizierte VPC-Lattice-Anforderungen

Examples for Signature Version 4 (SigV4) authenticated requests enthält Python- und Java-Beispiele, die zeigen, wie Sie das Signieren von Anforderungen mit und ohne benutzerdefinierte Interceptors durchführen können.

Verwenden von Signature Version 4 mit Amazon Translate

Verwenden von Signature Version 4 mit Amazon Translate zeigt, wie ein Python-Programm verwendet wird, um Authentifizierungsinformationen zu Amazon-Translate-Anforderungen hinzuzufügen. Das Beispiel macht eine POST-Anfrage, erstellt eine JSON-Struktur, die den zu übersetzenden Text im Text (Nutzlast) der Anfrage enthält, und übergibt Authentifizierungsinformationen in einem Authorization-Header.

Verwenden von Signature Version 4 mit Neptune

Example: Connecting to Neptune Using Python with Signature Version 4 Signing zeigt, wie signierte Anforderungen mithilfe von Python an Neptune gestellt werden. Dieses Beispiel enthält Varianten für die Verwendung eines Zugriffsschlüssels oder temporärer Anmeldeinformationen.

Signieren von HTTP-Anforderungen für S3 Glacier

In der exemplarischen Signaturberechnung für die Streaming-API werden Schritt für Schritt die Einzelheiten der Erstellung einer Signatur für „Upload-Archiv (POST-Archiv)“ beschrieben. Hierbei handelt es sich um eine der beiden Streaming-APIs in S3 Glacier.

Übermitteln von HTTP-Anforderungen an Amazon SWF

Unter Making HTTP Requests to Amazon SWF werden die Header-Inhalte für eine JSON-Anforderung für Amazon SWF gezeigt.

Signaturberechnung für Streaming-APIs in Amazon OpenSearch Service

Das Signieren einer Amazon OpenSearch Service-Suchanfrage mit dem AWS SDK for PHP Version 3 enthält ein Beispiel dafür, wie signierte HTTP-Anfragen an Amazon OpenSearch Service gesendet werden.

Beispielprojekte im AWS Beispiel-Repository

Die folgenden Beispielprojekte zeigen, wie Anfragen signiert werden, um REST-API-Anfragen an AWS Dienste mit gängigen Sprachen wie Python, Node.js, Java, C#, Go und Rust zu stellen.

Projekte mit Signaturversion 4a

Das Projekt sigv4-signing-examples bietet Beispiele dafür, wie Anfragen mit SigV4a signiert werden können, um Rest-API-Anfragen AWS-Services mit gängigen Sprachen wie Python, Node.js, Java, C#, Go und Rust zu stellen.

Das a-signing-examplessigv4-Projekt bietet Beispiele für das Signieren von API-Anfragen für mehrere Regionen, zum Beispiel Multi-Region-Access Points in Amazon S3.

Veröffentlichen auf AWS IoT Core

Python-Code zum Veröffentlichen AWS IoT Core mit dem HTTPS-Protokoll bietet Anleitungen zum Veröffentlichen von Nachrichten AWS IoT Core mithilfe des HTTPS-Protokolls und der AWS SigV4-Authentifizierung. Es hat zwei Referenzimplementierungen - eine in Python und eine in NodeJs.

.Net Framework-Anwendung zur Veröffentlichung AWS IoT Core unter Verwendung des HTTPS-Protokolls bietet Anleitungen zum Veröffentlichen von Nachrichten AWS IoT Core mithilfe des HTTPS-Protokolls und der AWS SigV4-Authentifizierung. Dieses Projekt beinhaltet auch eine .NET Core-äquivalente Implementierung.

Fehlerbehebung bei signierten Anfragen für AWS -APIs

> **Important**
>
> Sofern Sie die AWS SDKs oder CLI nicht verwenden, müssen Sie Code schreiben, um Signaturen zu berechnen, die Authentifizierungsinformationen in Ihren Anfragen enthalten. Die Berechnung der SigV4-Signatur kann ein komplexes Unterfangen sein, und wir empfehlen Ihnen, wann immer möglich die AWS -SDKs oder die CLI zu verwenden.

Wenn Sie Code entwickeln, der eine signierte Anfrage erstellt, erhalten Sie möglicherweise HTTP 403 `SignatureDoesNotMatch` von AWS-Services. Diese Fehler bedeuten, dass der Signaturwert in Ihrer HTTP-Anfrage AWS nicht mit der AWS-Service berechneten Signatur übereinstimmt. HTTP `Unauthorized` 401-Fehler werden zurückgegeben, wenn die Berechtigungen es dem Aufrufer nicht erlauben, die Anfrage zu stellen.

API-Anforderungen können in folgenden Fällen einen Fehler zurückgeben:

- Die API-Anforderung ist nicht signiert und die API-Anfrage verwendet die IAM-Authentifizierung.
- Die zum Signieren der Anfrage verwendeten IAM-Anmeldeinformationen sind falsch oder verfügen nicht über die erforderlichen Berechtigungen zum Aufrufen der API.
- Die Signatur der signierten API-Anfrage stimmt nicht mit der Signatur überein, die der AWS -Service berechnet hat.
- Der API-Anforderungs-Header ist falsch.

> **Note**
>
> Aktualisieren Sie Ihr Signaturprotokoll von AWS Signature Version 2 (SigV2) auf AWS Signature Version 4 (Sigv4), bevor Sie sich mit anderen Fehlerlösungen befassen. Services wie Amazon S3 und Regionen unterstützen die SIGv2-Signatur nicht mehr.

Mögliche Ursachen

- Fehler bei den Anmeldeinformationen
- Fehler bei der kanonischen Anfrage und beim Signieren der Zeichenfolge
- Fehler im Geltungsbereich der Anmeldeinformationen
- Fehler bei der Schlüsselsignierung

Fehler bei den Anmeldeinformationen

Stellen Sie sicher, dass die API-Anfrage mit SigV4 signiert ist. Wenn die API-Anfrage nicht signiert ist, erhalten Sie möglicherweise den folgenden Fehler: `Missing Authentication Token`. Fügen Sie die fehlende Signatur hinzu und senden Sie die Anfrage erneut.

Stellen Sie sicher, dass die Authentifizierungsdaten für den Zugriffsschlüssel und den geheimen Schlüssel korrekt sind. Wenn der Zugriffsschlüssel falsch ist, erhalten Sie möglicherweise die folgende Fehlermeldung: `Unauthorized`. Stellen Sie sicher, dass die zum Signieren der Anfrage verwendete Entität berechtigt ist, die Anfrage zu stellen. Details hierzu finden Sie unter Fehlerbehebung von Meldungen aufgrund Zugriffsverweigerung.

Fehler bei der kanonischen Anfrage und beim Signieren der Zeichenfolge

Wenn Sie die kanonische Anfrage in Schritt 2: Erstellen eines Hashes der kanonischen Anforderung oder Schritt 3: Erstellen einer zu signierenden Zeichenfolge falsch berechnet haben, schlägt der vom Service durchgeführte Signaturüberprüfungsschritt mit der Fehlermeldung fehl:

```
The request signature we calculated does not match the signature you provided
```

Wenn der AWS Dienst eine signierte Anfrage empfängt, berechnet er die Signatur neu. Wenn die Werte unterschiedlich sind, stimmen die Signaturen nicht überein. Vergleichen Sie die kanonische Anfrage und die Zeichenfolge mit Ihrer signierten Anfrage mit dem Wert in der Fehlermeldung. Ändern Sie den Signaturprozess, falls es Unterschiede gibt.

> **Note**
>
> Sie können auch überprüfen, ob Sie die Anfrage nicht über einen Proxy gesendet haben, der die Header oder die Anfrage ändert.

Example Beispiel für kanonische Anforderung

```
GET                                                    -------- HTTP method
/                                                      -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                       -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                          -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com    -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                           -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501       -------- Hash
 of the payload
```

Um zu überprüfen, ob der Geheimschlüssel mit der Zugriffsschlüssel-ID übereinstimmt, können Sie ihn mit einer bekannten funktionierenden Implementierung testen. Verwenden Sie beispielsweise ein AWS SDK oder die AWS CLI, um eine Anfrage zu stellen AWS.

Header der API-Anfrage

Stellen Sie sicher, dass der SigV4-Autorisierungs-Header, den Sie in Schritt 4: Berechnen der Signatur hinzugefügt haben, den richtigen Anmeldeinformationsschlüssel ähnlich dem folgenden enthält:

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

Wenn der Anmeldeinformationsschlüssel fehlt oder falsch ist, erhalten Sie möglicherweise die folgende Fehlermeldung: `Authorization header requires 'Credential' parameter. Authorization header requires 'Signature' parameter.`. Stellen Sie sicher, dass die SigV4-Autorisierungsanfrage auch das Anforderungsdatum enthält, indem Sie entweder den HTTP `Date`- oder den `x-amz-date`-Header verwenden.

Fehler im Geltungsbereich der Anmeldeinformationen

Der in Schritt 3: Erstellen einer zu signierenden Zeichenfolge erstellte Anmeldeinformationsbereich beschränkt eine Signatur auf ein bestimmtes Datum, eine bestimmte Region und einen bestimmten Service. Die Zeichenfolge hat das folgende Format:

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Wenn Sie SigV4a verwenden, ist die Region nicht im Gültigkeitsbereich der Anmeldeinformationen enthalten.

Datum

Wenn der Geltungsbereich der Anmeldeinformationen nicht dasselbe Datum wie der x-amz-date-Header angibt, schlägt der Schritt zur Signaturüberprüfung mit der folgenden Fehlermeldung fehl:

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Wenn die Anforderung einen Zeitpunkt in der Zukunft angibt, schlägt der Schritt zur Signaturüberprüfung mit der folgenden Fehlermeldung fehl:

```
Signature not yet current: date is still later than date
```

Wenn die Anfrage abgelaufen ist, schlägt der Schritt zur Signaturüberprüfung mit der folgenden Fehlermeldung fehl:

```
Signature expired: date is now earlier than date
```

Region

Wenn der Geltungsbereich der Anmeldeinformationen nicht dieselbe Region wie die Anfrage angibt, schlägt der Schritt zur Signaturüberprüfung mit der folgenden Fehlermeldung fehl:

```
Credential should be scoped to a valid Region, not region-code
```

Service

Wenn der Geltungsbereich der Anmeldeinformationen nicht denselben Service wie der host-Header angibt, schlägt der Schritt zur Signaturüberprüfung mit der folgenden Fehlermeldung fehl:

```
Credential should be scoped to correct service: 'service'
```

Abschlusszeichenfolge

Wenn der Geltungsbereich der Anmeldeinformationen nicht mit aws4_request endet, schlägt der Schritt zur Signaturüberprüfung mit der folgenden Fehlermeldung fehl:

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Fehler bei der Schlüsselsignierung

Fehler, die durch falsche Ableitung des Signaturschlüssels oder unsachgemäße Verwendung von Kryptografie verursacht werden, sind schwieriger zu beheben. Nachdem Sie überprüft haben, ob die kanonische Zeichenfolge und die zu signierende Zeichenfolge korrekt sind, können Sie auch überprüfen, ob eines der folgenden Probleme vorliegt:

- Der geheime Zugriffsschlüssel stimmt nicht mit der Zugriffsschlüssel-ID überein, die Sie angegeben haben.

- Es liegt ein Problem mit dem Code zur Schlüsselableitung vor.

Um zu überprüfen, ob der Geheimschlüssel mit der Zugriffsschlüssel-ID übereinstimmt, können Sie ihn mit einer bekannten funktionierenden Implementierung testen. Verwenden Sie beispielsweise ein AWS SDK oder das AWS CLI , um eine Anfrage zu stellen AWS. Beispiele finden Sie unter Anfordern von Signaturbeispielen

IAM-JSON-Richtlinienreferenz

Dieser Abschnitt enthält detaillierte Syntax, Beschreibungen und Beispiele für die Elemente, Variablen und Auswertungslogik von JSON-Richtlinien in IAM. Weitere allgemeine Informationen finden Sie unter Übersicht über JSON-Richtlinien.

Diese Referenz enthält folgende Abschnitte.

- IAM-JSON-Richtlinienelementreferenz – erfahren Sie mehr über die Elemente, die Sie beim Erstellen einer Richtlinie verwenden können. Zeigen Sie weitere Richtlinienbeispiele an und erfahren Sie mehr über Bedingungen, unterstützte Datentypen und ihre Verwendung in verschiedenen Services.
- Auswertungslogik für Richtlinien— In diesem Abschnitt werden AWS Anfragen beschrieben, wie sie authentifiziert werden und wie Richtlinien AWS verwendet werden, um den Zugriff auf Ressourcen zu bestimmen.
- Grammatik der IAM-JSON-Richtliniensprache : – In diesem Abschnitt wird die formale Grammatik der zum Erstellen von Richtlinien in IAM verwendeten Sprache vorgestellt.
- AWS verwaltete Richtlinien für Jobfunktionen – In diesem Abschnitt werden alle von AWS verwalteten Richtlinien aufgelistet, die den in der IT-Branche üblichen Auftragsfunktionen entsprechen. Verwenden Sie diese Richtlinien, um die zur Ausführung der in einer bestimmten Auftragsfunktion erwarteten Aufgaben erforderlichen Berechtigungen zu gewähren. Diese Richtlinien konsolidieren Berechtigungen für viele Services in einer einzigen Richtlinie.
- AWS Kontextschlüssel für globale Bedingungen— Dieser Abschnitt enthält eine Liste aller AWS globalen Bedingungsschlüssel, mit denen Sie die Berechtigungen in einer IAM-Richtlinie einschränken können.
- IAM- und AWS STS Bedingungskontextschlüssel— Dieser Abschnitt enthält eine Liste aller IAM- und AWS STS Bedingungsschlüssel, mit denen Sie die Berechtigungen in einer IAM-Richtlinie einschränken können.

- Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste — Dieser Abschnitt enthält eine Liste aller AWS API-Operationen, die Sie als Berechtigungen in einer IAM-Richtlinie verwenden können. Er enthält außerdem servicespezifische Bedingungsschlüssel, die zur Feinabstimmung der Anforderung herangezogen werden können.

IAM-JSON-Richtlinienelementreferenz

JSON-Richtliniendokumente bestehen aus Elementen. Die Elemente sind in der Reihenfolge aufgelistet, in der Sie in einer Richtlinie verwendet werden. Die Reihenfolge der Elemente ist unerheblich - Das Element `Resource` kann z. B. vor dem Element `Action` angeordnet sein. Sie müssen nicht alle `Condition`-Elemente in der Richtlinie angeben. Weitere Informationen zur allgemeinen Struktur und zum Zweck eines JSON-Richtliniendokuments finden Sie unter Übersicht über JSON-Richtlinien.

Einige Elemente der JSON-Richtlinie schließen sich gegenseitig. Das bedeutet, dass Sie keine Richtlinie erstellen können, in der beides verwendet wird. Sie können beispielsweise `Action` und `NotAction` nicht in der gleichen Richtlinienanweisung verwenden. Auch die Paare `Principal`/`NotPrincipal` und `Resource`/`NotResource` schließen sich gegenseitig.

Die in einer Richtlinie enthaltenen Elemente sind für jeden Service unterschiedlich und davon abhängig, welche Aktionen der Service bereitstellt, welche Ressourcentypen enthalten sind usw. Wenn Sie Richtlinien für einen bestimmten Service definieren, ist es hilfreich, sich auf Richtlinienbeispiele für diesen Service zu beziehen. Eine Liste aller Services mit IAM-Unterstützung und Links zu der Dokumentation für diese Services, in der IAM und Richtlinien behandelt werden, finden Sie unter AWS Dienste, die mit IAM funktionieren.

Wenn Sie eine JSON-Richtlinie erstellen oder bearbeiten, kann IAM eine Richtlinienvalidierung durchführen, um Ihnen beim Erstellen einer effektiven Richtlinie zu helfen. IAM identifiziert JSON-Syntaxfehler, während IAM Access Analyzer zusätzliche Richtlinienüberprüfungen mit Empfehlungen zur weiteren Verfeinerung Ihrer Richtlinien bietet. Weitere Informationen zur Richtlinienvalidierung finden Sie unter Validieren von IAM-Richtlinien. Weitere Informationen zu den Richtlinienvalidierungen von IAM Access Analyzer Richtlinien und Empfehlungen erhalten Sie unter Richtlinienvalidierung von IAM Access Analyzer.

Themen

- IAM-JSON-Richtlinienelemente: Version
- IAM-JSON-Richtlinienelemente: Id

- IAM-JSON-Richtlinienelemente: Statement
- IAM-JSON-Richtlinienelemente: Sid
- IAM-JSON-Richtlinienelemente: Effect
- AWS JSON-Richtlinienelemente: Principal
- AWS JSON-Richtlinienelemente: NotPrincipal
- IAM-JSON-Richtlinienelemente: Action
- IAM-JSON-Richtlinienelemente: NotAction
- IAM-JSON-Richtlinienelemente: Resource
- IAM-JSON-Richtlinienelemente: NotResource
- IAM-JSON-Richtlinienelemente: Condition
- IAM-Richtlinienelemente: Variablen und Tags
- IAM-JSON-Richtlinienelemente: Unterstützte Datentypen

IAM-JSON-Richtlinienelemente: Version

> **Hinweis zur Begriffsklärung**
>
> Dieses `Version`JSON Richtlinienelement unterscheidet sich von einer Richtlinienversion. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Andererseits wird eine Richtlinienversion erstellt, wenn Sie in IAM eine benutzerdefinierte, verwaltete Richtlinie bearbeiten. Die vorhandene Richtlinie wird von der geänderten Richtlinie nicht überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. Weitere Informationen zur Unterstützung mehrerer Versionen für verwaltete Richtlinien finden Sie unter the section called “Versioning von IAM-Richtlinien”.

Die `Version`-Richtlinienelemente legen die Sprachsyntaxregeln fest, die für die Verarbeitung einer Richtlinie verwendet werden sollen. Um alle verfügbaren Richtlinienfunktionen zu nutzen, fügen Sie in all Ihre Richtlinien das folgende `Version`-Element vor dem `Statement`-Element ein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

IAM unterstützt die folgenden `Version`-Elementwerte:

- `2012-10-17`. Hierbei handelt es sich um die aktuelle Version der Richtliniensprache. Sie sollte immer ein `Version`-Element einschließen mit `2012-10-17`. Andernfalls können Sie keine Funktionen verwenden, wie z. B. Richtlinienvariablen, die mit dieser Version eingeführt wurden.
- `2008-10-17`. Dies ist eine ältere Version der Richtliniensprache. Diese Version kann in älteren Richtlinien verwendet werden. Verwenden Sie diese Version nicht für neue Richtlinien oder wenn Sie vorhandene Richtlinien aktualisieren. Neuere Features, z. B. Richtlinienvariablen, funktionieren nicht mit Ihrer Richtlinie. Beispielsweise werden Variablen wie `${aws:username}` nicht als Variablen erkannt und stattdessen als literale Zeichenketten in der Richtlinie behandelt.

IAM-JSON-Richtlinienelemente: Id

Das `Id`-Element definiert eine optionale ID für die Richtlinie. Die Verwendung der ID in verschiedenen Services ist unterschiedlich. Die ID ist in ressourcenbasierten Richtlinien zulässig, nicht jedoch in identitätsbasierten Richtlinien.

Für Services, für die Sie ein `ID`-Element definieren können, empfehlen wir, dass Sie eine UUID (GUID) für den Wert verwenden oder dass Sie zur Sicherstellung der Eindeutigkeit eine UUID als Bestandteil der ID aufnehmen.

```
{
  "Version": "2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Für einige AWS Dienste (z. B. Amazon SQS oder Amazon SNS) ist dieses Element möglicherweise erforderlich und es gelten Eindeutigkeitsanforderungen. Servicespezifische Hinweise zur Erstellung von Richtlinien finden Sie in der Dokumentation für den entsprechenden Service.

IAM-JSON-Richtlinienelemente: Statement

Das Element `Statement` ist das Hauptelement einer Richtlinie. Dieses Element ist erforderlich. Das `Statement`-Element kann eine einzelne Anweisung oder ein Array von einzelnen Anweisungen enthalten. Jeder einzelne Anweisungsblock muss in geschweifte Klammern { } eingeschlossen sein. Bei mehreren Anweisungen muss das Array in eckige Klammern [ ] eingeschlossen sein.

```
"Statement": [{...},{...},{...}]
```

Das folgende Beispiel zeigt eine Richtlinie mit einem Array mit drei Anweisungen in einem `Statement`-Element. (Die Richtlinie gewährt Ihnen den Zugriff auf Ihre eigenen "Basisordner" in der Amazon S3-Konsole.) Die Richtlinie enthält die Variable `aws:username`, die während der Richtlinienauswertung durch den in der Anforderung enthaltenen Benutzernamen ersetzt wird. Weitere Informationen finden Sie unter Einführung.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
```

```
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

IAM-JSON-Richtlinienelemente: Sid

Sie können eine `Sid` (Kontoausweis-ID) als optionale Kennung für die Richtlinienerklärung angeben. Sie können jeder Anweisung in einem Statement-Array einen `Sid`-Wert zuweisen. Sie können den `Sid`-Wert als Beschreibung für die Richtlinienerklärung verwenden. In Services, bei denen Sie ein `ID`-Element angeben können, wie z. B. SQS und SNS, ist der `Sid`-Wert nur eine Sub-ID der ID des Richtliniendokuments. In IAM muss der `Sid`-Wert innerhalb einer JSON-Richtlinie eindeutig sein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

Das `Sid`-Element unterstützt ASCII-Großbuchstaben (A-Z), Kleinbuchstaben (a–z) und Ziffern (0–9).

In IAM ist `Sid` nicht im IAM-API veröffentlicht. Sie können keine bestimmte Anweisung basierend auf dieser ID abrufen.

> **Note**
>
> Für einige AWS Dienste (z. B. Amazon SQS oder Amazon SNS) ist dieses Element möglicherweise erforderlich und es gelten Eindeutigkeitsanforderungen. Servicespezifische Hinweise zur Erstellung von Richtlinien finden Sie in der Dokumentation für den entsprechenden Service.

IAM-JSON-Richtlinienelemente: Effect

Das Element `Effect` ist erforderlich und gibt an, ob die Anweisung eine Zugriffserlaubnis oder eine explizite Zugriffsverweigerung bewirkt. Gültige Werte für `Effect` sind `Allow` und `Deny`. Beim `Effect`-Wert ist die Groß- und Kleinschreibung zu beachten.

```
"Effect":"Allow"
```

Standardmäßig wird der Zugriff auf Ressourcen verweigert. Um den Zugriff auf eine Ressource zuzulassen, müssen Sie das `Effect`-Element auf `Allow` setzen. Um eine Zugriffserlaubnis zu überschreiben (z. B. um eine aktive Zugriffserlaubnis zu überschreiben), setzen Sie das Element `Effect` auf `Deny`. Weitere Informationen finden Sie unter Auswertungslogik für Richtlinien.

AWS JSON-Richtlinienelemente: Principal

Verwenden Sie das `Principal`-Element in einer ressourcebasierten JSON Richtlinie, um den Auftraggeber anzugeben, dem der Zugriff auf eine Ressource erlaubt oder verweigert wird.

Sie müssen das `Principal`-Element in ressourcenbasierten Richtlinien verwenden. Mehrere Services unterstützen ressourcenbasierte Richtlinien, einschließlich IAM. Der ressourcenbasierte IAM-Richtlinientyp ist eine Rollenvertrauensrichtlinie. Verwenden Sie in IAM-Rollen das `Principal`-Element in der Rollenvetrauensichtlinie, um anzugeben, wer diese Rolle übernehmen kann. Für den kontoübergreifenden Zugriff müssen Sie die 12-stellige ID des vertrauenswürdigen Kontos angeben. Informationen darüber, ob Auftraggeber in Konten außerhalb Ihrer Vertrauenszone (vertrauenswürdige Organisation oder Konto) Zugriff zur Annahme Ihrer Rollen haben, finden Sie unter Was ist IAM Access Analyzer?.

> **Note**
>
> Nachdem Sie die Rolle erstellt haben, können Sie das Konto zu „*“ ändern, damit jeder die Rolle übernehmen kann. Wenn Sie dies tun, empfehlen wir dringend, dass Sie den Zugriff

auf die Rolle mit anderen Mitteln begrenzen, wie z. B. mit einem `Condition`-Element, das den Zugriff auf bestimmte IP-Adressen beschränkt. Schränken Sie den Zugriff auf Ihre Rolle unbedingt ein!

Andere Beispiele für Ressourcen, die ressourcenbasierte Richtlinien unterstützen, sind ein Amazon-S3-Bucket oder einen AWS KMS key.

Sie können das Element `Principal` nicht in einer identitätsbasierten Richtlinie verwenden. Identitätsbasierte Richtlinien sind Berechtigungsrichtlinien, die Sie an eine IAM-Identität anfügen können, wie z. B. -Benutzer, -Rollen oder -Gruppen. In diesen Richtlinien wird der Prinzipal von der Identität impliziert, der der Richtlinie angefügt ist.

Themen

- Angeben eines Auftraggebers
- AWS-Konto Schulleiter
- IAM-Rollenauftraggeber
- Rollensitzungsgsprinzipale
- IAM-Benutzerprinzipal
- Prinzipale von IAM Identity Center
- AWS STS Prinzipale für föderierte Benutzersitzungen
- AWS Dienstprinzipale
- AWS Serviceprinzipale in Opt-in-Regionen
- Alle Prinzipale
- Weitere Informationen

Angeben eines Auftraggebers

Sie geben einen Prinzipal in dem `Principal`-Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln an, die Prinzipale unterstützen.

Sie können einen der folgenden Auftraggeber in einer Richtlinie angeben:

- AWS-Konto und Root-Benutzer
- IAM-Rollen

- Rollensitzungen
- IAM-Benutzer
- Verbundbenutzersitzungen
- AWS Dienste
- Alle Prinzipale

Sie können eine Benutzergruppe nicht als Prinzipal in einer Richtlinie (z. B. einer ressourcenbasierten Richtlinie) identifizieren, da sich Gruppen auf Berechtigungen und nicht auf die Authentifizierung beziehen, während Prinzipale authentifizierte IAM-Entitäten sind.

Sie können mehrere Auftraggeber für jeden der Auftraggebertypen in den folgenden Abschnitten mithilfe eines Arrays angeben. Arrays können einen oder mehrere Werte enthalten. Wenn Sie mehr als einen Auftraggeber im Element angeben, erteilen Sie jedem Auftraggeber Berechtigungen. Dies ist ein logisches `OR` und kein logisches `AND`, da Sie jeweils als ein Auftraggeber authentifiziert sind. Wenn Sie mehr als einen Wert angeben, verwenden Sie eckige Klammern (`[` und `]`) und trennen Sie jeden Eintrag für das Array durch Kommas. Die folgende Beispielrichtlinie definiert Berechtigungen für das 123456789012-Konto oder das 555555555555-Konto.

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> Sie können keinen Platzhalter verwenden, um einen Teil eines Namens oder eines ARNs zu ersetzen.

AWS-Konto Schulleiter

Sie können AWS-Konto Bezeichner im `Principal` Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln angeben, die Prinzipale unterstützen. Dadurch wird die Autorität des Kontos delegiert. Wenn Sie den Zugriff auf ein anderes Konto zulassen, muss ein Administrator in diesem Konto dann Zugriff auf eine Identität (IAM-Benutzer oder -Rolle) in diesem Konto gewähren.

Wenn Sie eine angeben AWS-Konto, können Sie den Konto-ARN (arn:aws:iam:: *Account-ID:root*) oder eine verkürzte Form verwenden, die aus dem Präfix gefolgt von der Konto-ID besteht. `"AWS":`

Wenn die Konto-ID beispielsweise `123456789012` lautet, können Sie eine der folgenden Methoden verwenden, um das betreffende Konto im Element `Principal` anzugeben:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

Mit dem Konto-ARN und der verkürzten Konto-ID verhält es sich genauso. Beide delegieren Berechtigungen für das Konto. Wenn das Konto ARN im `Principal`-Element verwendet wird, werden die Berechtigungen nicht nur auf den Stamm-Benutzer des Kontos beschränkt.

> **Note**
>
> Wenn Sie eine ressourcenbasierte Richtlinie speichern, die die verkürzte Konto-ID enthält, konvertiert der Dienst sie möglicherweise in den Haupt-ARN. Dies ändert nichts an der Funktionalität der Richtlinie.

Einige AWS Dienste unterstützen zusätzliche Optionen für die Angabe eines Kontoprinzipals. Bei Amazon S3 können Sie beispielsweise eine kanonische Benutzer-ID in folgendem Format angeben:

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Sie können mithilfe eines Arrays auch mehr als eine AWS-Konto(oder kanonische Benutzer-ID) als Prinzipal angeben. Beispielsweise können Sie mit allen drei Methoden einen Prinzipal in einer Bucket-Richtlinie angeben.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
```

```
}
```

IAM-Rollenauftraggeber

Sie können IAM-Rollenprinzipal-ARNs im `Principal`-Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln, die Prinzipale unterstützen. IAM-Rollen sind Identitäten. In IAM sind Identitäten Ressourcen, denen Sie Berechtigungen zuweisen können. Rollen vertrauen einer anderen authentifizierten Identität, um diese Rolle zu übernehmen. Dies beinhaltet einen Prinzipal in AWS oder einem Benutzer eines externen Identitätsanbieters (IdP). Wenn ein Prinzipal oder eine Identität eine Rolle übernimmt, erhalten sie temporäre Sicherheitsanmeldeinformationen mit den Berechtigungen der angenommenen Rolle. Wenn sie diese Sitzungsanmeldedaten für die Ausführung von Vorgängen verwenden AWS, werden sie zu einer Rolle als Sitzungsprinzipal.

IAM-Rollen sind Identitäten, die in IAM existieren. Rollen vertrauen einer anderen authentifizierten Identität, z. B. einem Prinzipal in einem externen Identitätsanbieter AWS oder einem Benutzer von einem externen Identitätsanbieter. Wenn ein Prinzipal oder eine Identität eine Rolle übernimmt, erhält er temporäre Sicherheitsanmeldeinformationen. Sie können diese Anmeldeinformationen dann als Rollensitzungsprinzipal verwenden, um Vorgänge in AWS durchzuführen.

Wenn Sie einen Rollenprinzipal in einer ressourcenbasierten Richtlinie angeben, sind die effektiven Berechtigungen für den Prinzipal durch alle Richtlinientypen begrenzt, die Berechtigungen für die Rolle einschränken. Dies umfasst Sitzungssrichtlinien und Berechtigungsgrenzen. Weitere Informationen zur Auswertung der effektiven Berechtigungen für eine Rollensitzung finden Sie unter Auswertungslogik für Richtlinien.

Um die Rolle ARN im `Principal`-Element anzugeben, verwenden Sie das folgende Format:

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Wenn Ihr `Principal`-Element in einer Vertrauensrichtlinie einer Rolle einen ARN enthält, der auf eine bestimmte IAM-Rolle verweist, wird dieser ARN beim Speichern der Richtlinie in die eindeutige Auftraggeber-ID der Rolle umgewandelt. Auf diese Weise wird das Risiko reduziert, dass jemand seine Berechtigungen durch Entfernen und Neuerstellen der Rolle erweitert. Normalerweise wird diese ID nicht in der Konsole angezeigt, da IAM bei der Anzeige der Vertrauensrichtlinie eine Rückumwandlung zum Rollen-ARN verwendet. Wenn Sie jedoch die Rolle löschen, wird die Beziehung aufgehoben. Die Richtlinie wird nicht mehr

angewendet, selbst wenn Sie die Rolle neu erstellen, da die Auftraggeber-ID der neuen Rolle nicht mit der in der Vertrauensrichtlinie gespeicherten ID übereinstimmt. In diesem Fall wird die Prinzipal-ID in ressourcenbasierten Richtlinien angezeigt, da sie nicht mehr einem gültigen ARN zugeordnet werden AWS kann. Daher müssen Sie die Rolle bearbeiten, um die nunmehr falsche Prinzipal-ID mit dem richtigen ARN zu ersetzen, wenn Sie eine mit einem `Principal`-Element einer Vertrauensrichtlinie verknüpfte Rolle löschen und neu erstellen. Der ARN wird beim Speichern der Richtlinie erneut in die neue Auftraggeber-ID der Rolle umgewandelt.

Alternativ können Sie den Rollenprinzipal als Prinzipal in einer ressourcenbasierten Richtlinie angeben oder erstellen Sie eine Richtlinie mit breiter Berechtigung die `aws:PrincipalArn`-Bedingungsschlüssel benutzt. Wenn Sie diesen Schlüssel verwenden, erteilen Sie dem Rollensitzungsprinzipalen die Berechtigungen basierend auf dem übernommenen ARN der Rolle und nicht dem ARN der resultierenden Sitzung. Da Bedingungsschlüssel-ARNs AWS nicht in IDs konvertiert werden, bleiben die dem Rollen-ARN gewährten Berechtigungen bestehen, wenn Sie die Rolle löschen und dann eine neue Rolle mit demselben Namen erstellen. Identitätsbasierte Richtlinientypen, wie z. B. Berechtigungsgrenzen oder Sitzungsrichtlinien, schränken die gewährten Berechtigungen nicht ein, indem sie den `aws:PrincipalArn`-Bedingungsschlüssel mit einem Platzhalter (*) im `Principal`-Element verwenden, es sei denn, die identitätsbasierten Richtlinien enthalten eine ausdrückliche Verweigerung.

Rollensitzungsgsprinzipale

Sie können Rollensitzungen im `Principal`-Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln, die Prinzipale unterstützen, angeben. Wenn ein Prinzipal oder eine Identität eine Rolle übernimmt, erhalten sie temporäre Sicherheitsanmeldeinformationen mit den Berechtigungen der angenommenen Rolle. Wenn sie diese Sitzungsanmeldedaten für die Ausführung von Vorgängen verwenden AWS, werden sie zu Rollen-Sitzungsprinzipalen.

Das Format, das Sie für einen Rollensitzungsprinzipal verwenden, hängt von dem AWS STS Vorgang ab, mit dem die Rolle übernommen wurde.

Darüber hinaus können Administratoren einen Prozess entwerfen, um zu steuern, wie Rollensitzungen ausgegeben werden. Sie können beispielsweise eine Ein-Klick-Lösung für ihre Benutzer bereitstellen, die einen vorhersehbaren Sitzungsnamen erstellt. Wenn Ihr Administrator dies tut, können Sie Rollensitzungsprinzipale in Ihren Richtlinien oder Bedingungsschlüsseln verwenden. Andernfalls können Sie die Rollen-ARN als Prinzipal im `aws:PrincipalArn`-Bedingungsschlüssel

angeben. Wie Sie die Rolle als Prinzipal angeben, kann die effektiven Berechtigungen für die resultierend Sitzung ändern. Weitere Informationen finden Sie unter IAM-Rollenauftraggeber.

Sitzungsauftraggeber mit übernommener Rolle

Ein Sitzungsprinzipal mit angenommener Rolle ist ein Sitzungsprinzipal, der sich aus der Verwendung des Vorgangs ergibt. AWS STS `AssumeRole` Weitere Informationen darüber, welche Prinzipale bei dieser Operation eine Rolle übernehmen können, finden Sie unter Vergleich der AWS STS API-Operationen.

Um den ARN mit angenommener Rollensitzung im `Principal`-Element anzugeben, verwenden Sie das folgende Format:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Wenn Sie eine Sitzung mit übernommener Rolle in einem `Principal`-Element angeben, können Sie keinen Platzhalter „*“ verwenden, der "alle Sitzungen" bedeutet. Auftraggeber müssen immer eine bestimmte Sitzung benennen.

OIDC-Sitzungsprinzipale

Ein OIDC-Sitzungsprinzipal ist ein Sitzungsprinzipal, der sich aus der Verwendung des Vorgangs ergibt. AWS STS `AssumeRoleWithWebIdentity` Sie können einen externen OIDC-Anbieter (IdP) verwenden, um sich anzumelden, und dann mit diesem Vorgang eine IAM-Rolle annehmen. Dies nutzt den Identitätsverbund und gibt eine Rollensitzung aus. Weitere Informationen darüber, welche Prinzipale bei dieser Produktion eine Rolle übernehmen können, finden Sie unter Vergleich der AWS STS API-Operationen.

Wenn Sie eine Rolle von einem OIDC-Anbieter ausgeben, erhalten Sie diesen speziellen Sitzungsprinzipaltyp, der Informationen über den OIDC-Anbieter enthält.

Verwenden Sie diesen Prinziptyp in Ihrer Richtlinie, um den Zugriff basierend auf dem vertrauenswürdigen Web-Identitätsanbieter zuzulassen oder abzulehnen. Verwenden Sie das folgende Format, um den ARN der OIDC-Rollensitzung im `Principal` Element einer Rollenvertrauensrichtlinie anzugeben:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

SAML-Sitzungsprinzipale

Ein SAML-Sitzungsprinzipal ist ein Sitzungsprinzipal, der sich aus der Verwendung des Vorgangs ergibt. AWS STS `AssumeRoleWithSAML` Sie können sich mit einem externen SAML-Identitätsanbieter (IdP) anmelden und dann mit dieser Produktion eine IAM-Rolle übernehmen. Dies nutzt den Identitätsverbund und gibt eine Rollensitzung aus. Weitere Informationen darüber, welche Prinzipale bei dieser Produktion eine Rolle übernehmen können, finden Sie unter Vergleich der AWS STS API-Operationen.

Wenn Sie eine Rolle von einem SAML-Identitätsanbieter ausstellen, erhalten Sie diesen speziellen Typ von Vortrasprinzipal, der Informationen über den SAML-Identitätsanbieter enthält.

Verwenden Sie diesen Prinziptyp in Ihrer Richtlinie, um den Zugriff basierend auf dem vertrauenswürdigen SAML-Identitätsanbieter zuzulassen oder abzulehnen. Um dem ARN der SAML-Identitäts-Rollensitzung im `Principal`-Element eine Rollenvetrauensrichtlinie zu geben, verwenden Sie das folgende Format:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

IAM-Benutzerprinzipal

Sie können IAM-Benutzer im `Principal`-Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln angeben, die Prinzipale unterstützen.

> **Note**
>
> Bei einem `Principal`-Element, bei dem Benutzernamen Teil des Amazon-Ressourcenname(ARN) ist, wird zwischen Groß- und Kleinschreibung unterschieden.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
```

```
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Wenn Sie Benutzer in einem `Principal`-Element angeben, können Sie keinen Platzhalter (*) für "alle Benutzer" verwenden. Auftraggeber müssen stets bestimmter Benutzer angeben.

> **Important**
>
> Wenn Ihr `Principal`-Element in einer Rollenvetrauensichtlinie einen ARN enthält, der auf einen bestimmten IAM-Benutzer verweist, wird dieser ARN beim Speichern der Richtlinie in die eindeutige Auftraggeber-ID des Benutzers umgewandelt. Auf diese Weise wird das Risiko reduziert, dass jemand seine Berechtigungen durch Entfernen und Neuerstellen des Benutzers erweitert. Normalerweise wird diese ID nicht in der Konsole angezeigt, da bei der Anzeige der Vertrauensrichtlinie auch eine Rückumwandlung in die Benutzer-ARN erfolgt. Wenn Sie jedoch den Benutzer löschen, wird die Beziehung aufgehoben. Die Richtlinie wird dann nicht mehr angewendet, selbst wenn Sie den Benutzer neu erstellen. Dies liegt daran, dass die Auftraggeber-ID des neuen Benutzers nicht mit der in der Vertrauensrichtlinie gespeicherten ID übereinstimmt. In diesem Fall wird die Prinzipal-ID in ressourcenbasierten Richtlinien angezeigt, da sie nicht mehr einem gültigen ARN zugeordnet werden AWS kann. Daher müssen Sie die Rolle bearbeiten, um die nunmehr falsche Auftraggeber-ID durch den richtigen ARN zu ersetzen, wenn Sie einen mit einem `Principal`-Element einer Vertrauensrichtlinie verknüpften Benutzer löschen und neu erstellen. IAM wandelt beim Speichern der Richtlinie den ARN erneut in die neue Haupt-ID des Benutzers um.

Prinzipale von IAM Identity Center

In IAM Identity Center muss der Prinzipal in einer ressourcenbasierten Richtlinie als AWS-Konto -Prinzipal definiert werden. Um Zugriff anzugeben, verweisen Sie auf den Rollen-ARN der im Bedingungsblock festgelegten Berechtigung. Einzelheiten finden Sie unter Referencing permission sets in resource policies, Amazon EKS, and AWS KMS im Benutzerhandbuch zu IAM Identity Center.

AWS STS Prinzipale für föderierte Benutzersitzungen

Sie können Verbundbenutzersitzungen im `Principal`-Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln angeben, die Prinzipale unterstützen.

> **Important**
>
> AWS empfiehlt, AWS STS Verbundbenutzersitzungen nur dann zu verwenden, wenn dies erforderlich ist, z. B. wenn Root-Benutzerzugriff erforderlich ist. Stattdessen, Verwenden von Rollen, um Berechtigungen zu delegieren.

Ein AWS STS föderierter Benutzersitzungsprinzipal ist ein Sitzungsprinzipal, der sich aus der Verwendung des AWS STS `GetFederationToken` Vorgangs ergibt. In diesem Fall verwendet AWS STS Identitätsverbund als Methode, um temporäre Zugriffstoken zu erhalten, anstatt IAM-Rollen zu verwenden.

In Root-Benutzer des AWS-Kontos können AWS sich IAM-Benutzer oder ein Benutzer mit langfristigen Zugriffsschlüsseln authentifizieren. Weitere Informationen zum Verbünden von Prinzipalen mittles dieser Produktion finden Sie unter Vergleich der AWS STS API-Operationen.

- IAM-Verbundbenutzer - Ein IAM-Benutzer verbündet, mithilfe einer `GetFederationToken`-Produktion, die zu einem Prinzipal für Verbundbenutzersitzungen für diesen IAM-Benutzer führt.
- Stamm-Verbundbenutzer - Ein Root-Benutzer verbündet mithilfe einer `GetFederationToken`-Produktion, die zu einem Prinzipal für Verbundbenutzer-Sitzungen für diesen Root-Benutzer führt.

Wenn ein IAM-Benutzer oder Root-Benutzer temporäre Anmeldeinformationen für diesen Vorgang anfordert, beginnt er eine temporäre Verbundbenutzersitzung. AWS STS Der ARN dieser Sitzung basiert auf der ursprünglichen Identität, die verbunden wurde.

Um den ARN der Verbundbenutzersitzung im `Principal`-Element anzugeben, verwenden Sie das folgende Format:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWS Dienstprinzipale

Sie können AWS Dienste als `Principal` Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln angeben, die Prinzipale unterstützen. Ein Service-Prinzipal ist eine Kennung für einen Service.

IAM-Rollen, die von einem Dienst übernommen werden können, werden als AWS Servicerollen bezeichnet. Service-Rollen müssen eine Vertrauensrichtlinie enthalten. Vertrauensrichtlinien sind

ressourcenbasierte Richtlinien, die einer Rolle zugeordnet sind. Sie definiert, welche Auftraggeber die Rolle übernehmen können. Einige Service-Rollen haben vordefinierte Vertrauensrichtlinien. In einigen Fällen müssen Sie jedoch den Dienstauftraggeber in der Vertrauensrichtlinie angeben. Der Dienstprinzipal in einer IAM-Richtlinie kann das nicht sein. `"Service": "*"`

Der Bezeichner für einen Dienstauftraggeber enthält den Servicenamen und hat normalerweise das folgende Format:

service-name.amazonaws.com

Der Dienstauftraggeber wird durch den Service definiert. Sie können den Dienstauftraggeber für einige Services finden, indem Sie AWS Dienste, die mit IAM funktionieren öffnen, überprüfen, ob der Service in der Spalte Service-linked role (Serviceverknüpfte Rolle) Yes (Ja) hat, und den Link Yes (Ja) öffnen, um die Dokumentation zu serviceverknüpften Rollen für diesen Service anzuzeigen. Suchen Sie den Abschnitt Service-Linked Role Permissions (Berechtigungen von serviceverknüpften Rollen) für diesen Service, um den Dienstauftraggeber anzuzeigen.

Das folgende Beispiel zeigt eine Richtlinie, die einer Service-Rolle angefügt werden kann. Diese Richtlinie ermöglicht zwei Services, Amazon ECS und Elastic Load Balancing, die Übernahme der Rolle. Die Services können dann sämtliche Aufgaben ausführen, zu denen die Rolle infolge der zugewiesenen Berechtigungsrichtlinie berechtigt ist (nicht dargestellt). Geben Sie zur Angabe von mehreren Dienstauftraggeber nicht zwei `Service`-Elemente an. Sie können nur ein Element angeben. Verwenden Sie stattdessen ein Array mit mehreren Dienstauftraggeber als Wert eines einzelnen `Service`-Elements.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWS Serviceprinzipale in Opt-in-Regionen

Sie können Ressourcen in mehreren AWS Regionen bereitstellen, und für einige dieser Regionen müssen Sie sich entscheiden. Eine vollständige Liste der Regionen, für die Sie sich anmelden müssen, finden Sie im Allgemeine AWS-ReferenzLeitfaden unter AWS Regionen verwalten.

Wenn ein AWS Dienst in einer Opt-in-Region eine Anfrage innerhalb derselben Region stellt, wird das Format des Dienstprinzipalnamens als die nicht regionalisierte Version seines Dienstprinzipalnamens identifiziert:

service-name.amazonaws.com

Wenn ein AWS Dienst in einer Opt-in-Region eine regionsübergreifende Anfrage an eine andere Region stellt, wird das Format des Dienstprinzipalnamens als die regionalisierte Version seines Dienstprinzipalnamens identifiziert:

service-name.*{region}*.amazonaws.com

Angenommen, Sie haben ein Amazon-SNS-Thema in der Region `ap-southeast-1` und einen Amazon-S3-Bucket in der Opt-in-Region `ap-east-1`. Sie möchten S3-Bucket-Benachrichtigungen konfigurieren, um Nachrichten im SNS-Thema zu veröffentlichen. Damit der S3-Service Nachrichten im SNS-Thema posten kann, müssen Sie dem S3-Serviceprinzipal über die ressourcenbasierte Zugriffsrichtlinie des Themas die Berechtigung `sns:Publish` erteilen.

Wenn Sie in der Themenzugriffsrichtlinie die nicht regionalisierte Version des S3-Serviceprinzipals `s3.amazonaws.com` angeben, schlägt die `sns:Publish`-Anfrage vom Bucket an das Thema fehl. Im folgenden Beispiel wird der nicht regionalisierte S3-Serviceprinzipal im `Principal`-Richtlinienelement der SNS-Themenzugriffsrichtlinie angegeben.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Da sich der Bucket in einer Opt-In-Region befindet und die Anfrage außerhalb dieser Region gestellt wurde, wird der regionalisierte Name des S3-Serviceprinzipals angezeigt: `s3.ap-east-1.amazonaws.com`. Sie müssen den regionalisierten Dienstprinzipalnamen verwenden, wenn ein AWS Dienst in einer Opt-in-Region eine Anfrage an eine andere Region sendet. Wenn Sie den regionalisierten Namen des Serviceprinzipals angegeben haben und der Bucket eine `sns:Publish`-Anfrage an das SNS-Thema in einer anderen Region stellt, ist die Anfrage erfolgreich. Im folgenden Beispiel wird der regionalisierte S3-Serviceprinzipal im `Principal`-Richtlinienelement der SNS-Themenzugriffsrichtlinie angegeben.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Ressourcenrichtlinien oder auf Serviceprinzipalen basierende Zulassungslisten für regionsübergreifende Anfragen von einer Opt-In-Region an eine andere Region sind nur erfolgreich, wenn Sie den regionalisierten Serviceprinzipalnamen angeben.

> **Note**
>
> Für Vertrauensrichtlinien von IAM-Rollen empfehlen wir, den nicht regionalisierten Serviceprinzipalnamen zu verwenden. IAM-Ressourcen sind global, daher kann dieselbe Rolle in jeder Region verwendet werden.

Alle Prinzipale

Sie können ein Platzhalterzeichen (*) verwenden, um alle Prinzipale im `Principal`-Element einer ressourcenbasierten Richtlinie oder in Bedingungsschlüsseln, die Prinzipale unterstützen, anzugeben. Ressourcenbasierte Richtlinien Erteilungs-Berechtigungen und Bedingungsschlüssel werden verwendet, um die Bedingungen einer Richtlinienanweisung einzuschränken.

> **Important**
>
> Es wird ausdrücklich empfohlen, keinen Platzhalter (*) im `Principal`-Element einer ressourcenbasierten Richtlinie mit `Allow`-Effekt zu verwenden, es sei denn, Sie beabsichtigen, öffentlichen oder anonymen Zugang zu gewähren. Andernfalls geben Sie die beabsichtigten Prinzipale, Services oder AWS -Konten im `Principal`-Element an und schränken dann den Zugriff im `Condition`-Element weiter ein. Dies gilt insbesondere für Vertrauensrichtlinien für IAM-Rollen, da sie es anderen Prinzipalen erlauben, Prinzipale in Ihrem Konto zu werden.

Für ressourcenbasierte Richtlinien wird durch die Verwendung eines Platzhalters (*) mit einem `Allow`-Effekt der Zugriff auf alle Benutzer, einschließlich anonymer Benutzer (öffentlicher Zugriff), gewährt. Für IAM-Benutzer- und Rollenprinzipale innerhalb Ihres Kontos sind keine weiteren Berechtigungen erforderlich. Für Prinzipale in anderen Konten müssen sie auch identitätsbasierte Berechtigungen in ihrem Konto haben, mit denen sie auf Ihre Ressource zugreifen können. Dies wird als kontenübergreifender Zugriff bezeichnet.

Für anonyme Benutzer sind die folgenden Elemente gleichwertig:

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Sie können keinen Platzhalter verwenden, um einen Teil eines Namens oder eines ARNs zu ersetzen.

Das folgende Beispiel zeigt eine ressourcenbasierte Richtlinie, die anstelle von Spezifizieren von `NotPrincipal` mit `Deny` verwendet werden kann, um alle Prinzipale mit Ausnahme der im Element `Condition` angegebenen ausdrücklich abzulehnen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME/*",
        "arn:aws:s3:::BUCKETNAME"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

Weitere Informationen

Weitere Informationen finden Sie hier:

- Beispiele für Bucket-Richtlinien im Benutzerhandbuch für Amazon Simple Storage Service
- Beispielrichtlinien für Amazon SNS im Entwicklerhandbuch für Amazon Simple Notification Service
- Beispiele für Amazon SQS Richtlinien im Entwicklerhandbuch für Amazon Simple Queue Service
- Schlüsselrichtlinien im Entwicklerhandbuch für AWS Key Management Service
- Kontokennungen in der Allgemeine AWS-Referenz
- OIDC-Föderation

AWS JSON-Richtlinienelemente: NotPrincipal

Sie können das `NotPrincipal` Element verwenden, um allen Prinzipalen den Zugriff zu verweigern, mit Ausnahme des IAM-Benutzers, des Verbundbenutzers, der IAM-Rolle, des AWS Dienstes oder eines anderen Prinzipals, der im Element angegeben ist. AWS-Konto`NotPrincipal`

Sie können es in ressourcenbasierten Richtlinien für einige AWS Dienste verwenden, einschließlich VPC-Endpoints. Ressourcenbasierte Richtlinien sind Richtlinien, die Sie direkt in eine Ressource einbinden. Sie können das `NotPrincipal`-Element weder in einer identitätsbasierten IAM-Richtlinie noch in einer IAM-Rollenvertrauensrichtlinie verwenden.

`NotPrincipal` muss mit `"Effect":"Deny"` verwendet werden. Die Verwendung mit `"Effect":"Allow"` wird nicht unterstützt.

> **Important**
>
> Nur sehr wenige Szenarien erfordern die Verwendung von `NotPrincipal`. Wir empfehlen Ihnen, andere Autorisierungsoptionen zu erkunden, bevor Sie sich für die Verwendung von `NotPrincipal` entscheiden. Wenn Sie `NotPrincipal` verwenden, kann die Fehlerbehebung bei den Auswirkungen mehrerer Richtlinientypen schwierig sein. Wir empfehlen, stattdessen den `aws:PrincipalArn`-Kontextschlüssel mit ARN-Bedingungsoperatoren zu verwenden. Weitere Informationen finden Sie unter Alle Prinzipale.

Spezifizieren von **`NotPrincipal`** mit **`Deny`**

Wenn Sie `NotPrincipal` mit `Deny` verwenden, müssen Sie auch den Konto-ARN des zugelassenen Auftraggebers angeben. Andernfalls könnte durch die Richtlinie der Zugriff auf das gesamte Konto mit dem Auftraggeber verweigert werden. Abhängig vom Service, den Sie in Ihre Richtlinien einschließen, validiert AWS möglicherweise erst das Konto und dann den Benutzer. Wenn ein Benutzer mit übernommener Rolle (jemand, der eine Rolle verwendet) bewertet wird, validieren Sie AWS möglicherweise zuerst das Konto, dann die Rolle und dann den Benutzer mit der übernommenen Rolle. Der Benutzer mit übernommener Rolle wird anhand des Namens der Rollensitzung identifiziert, die bei der Übernahme der Rolle durch den Benutzer angegeben wurde. Aus diesem Grund empfehlen wir dringend, dass Sie den ARN für das Konto eines Benutzers oder sowohl den ARN einer Rolle als auch den ARN für das Konto, das die Rolle enthält, mit aufnehmen.

Important

Verwenden Sie keine ressourcenbasierten Richtlinienanweisungen, die ein `NotPrincipal`-Richtlinienelement mit einer `Deny`-Wirkung für IAM-Benutzer oder -Rollen enthalten, denen eine Richtlinie mit Berechtigungsgrenzen angefügt ist. Das `NotPrincipal`-Element mit `Deny`-Wirkung lehnt immer jeden IAM-Prinzipal ab, an den eine Richtlinie zur Berechtigungsgrenze angefügt ist, unabhängig von den im `NotPrincipal`-Element angegebenen Werten. Dies führt dazu, dass einige IAM-Benutzer oder -Rollen, die andernfalls Zugriff auf die Ressource hätten, den Zugriff verlieren. Wir empfehlen Ihnen, Ihre ressourcenbasierten Richtlinien dahingehend zu ändern, dass Sie den Bedingungsoperator `ArnNotEquals` mit dem `aws:PrincipalArn`-Kontextschlüssel verwenden, um den Zugriff zu begrenzen, anstatt das `NotPrincipal`-Element. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten.

Note

Es hat sich bewährt, die ARNs für das Konto in die Richtlinie aufzunehmen. Einige Services erfordern den Konto-ARN, auch wenn dies nicht immer erforderlich ist. Alle vorhandenen Richtlinien ohne den erforderlichen ARN bleiben funktionsfähig, neue Richtlinien aber, die diese Services enthalten, müssen die Anforderung erfüllen. IAM verfolgt diese Services nicht und empfiehlt daher, dass Sie diese immer in den Konto-ARN einschließen.

Die folgenden Beispiele zeigen die effektive Verwendung von `NotPrincipal` und `"Effect": "Deny"` in derselben Richtlinienanweisung.

Example Beispiel eines IAM-Benutzers in demselben oder einem anderen Konto

Im folgenden Beispiel wird allen Hauptbenutzern mit Ausnahme des Benutzers Bob in AWS-Konto 444455556666 der Zugriff auf eine Ressource ausdrücklich verweigert. Beachten Sie, dass es sich bewährt hat, dass das `NotPrincipal` Element den ARN sowohl des Benutzers Bob als auch des Benutzers enthält AWS-Konto , zu dem Bob gehört (`arn:aws:iam::444455556666:root`). Wenn das `NotPrincipal` Element nur Bobs ARN enthielt, könnte die Richtlinie dazu führen, AWS-Konto dass der Zugriff auf das Element, das den Benutzer Bob enthält, explizit verweigert wird. In einen Fällen dürfen die Berechtigungen eines Benutzers nicht umfangreicher als die des übergeordneten Kontos sein. Wenn folglich dem Konto von Bob der Zugriff explizit verweigert wird, kann Bob möglicherweise nicht auf die Ressource zugreifen.

Dieses Beispiel funktioniert wie beabsichtigt, wenn es Teil einer Richtlinienanweisung in einer ressourcenbasierten Richtlinie ist, die einer Ressource entweder in derselben oder einer anderen AWS-Konto (nicht 444455556666) zugeordnet ist. Dieses Beispiel allein gewährt Bob noch keinen Zugriff, Bob wird lediglich aus der Liste der Auftraggeber ausgenommen, deren Zugriffsberechtigung explizit verweigert wird. Um Bob den Zugriff auf die Ressource zu gewähren, muss eine andere Richtlinienanweisung mithilfe von `"Effect": "Allow"` die Zugriffsberechtigung explizit erteilen.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:iam::444455556666:user/Bob",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::BUCKETNAME",
            "arn:aws:s3:::BUCKETNAME/*"
        ]
    }]
}
```

Example Beispiel einer IAM-Rolle in demselben oder einem anderen Konto

Im folgenden Beispiel wird allen Hauptbenutzern mit Ausnahme des in 444455556666 genannten Benutzers mit übernommener Rolle ausdrücklich der Zugriff auf eine Ressource verweigert. cross-account-audit-app AWS-Konto Als bewährte Methode enthält das `NotPrincipal` Element den ARN des Benutzers (cross-account-audit-app) mit der angenommenen Rolle, die Rolle (cross-account-read-only-role) und die Rolle, zu der AWS-Konto die Rolle gehört (444455556666). Wenn der ARN der Rolle nicht im `NotPrincipal`-Element enthalten wäre, würde dies vermutlich zu einer Zugriffsverweigerung der Rolle durch die Richtlinie führen. Wenn der ARN des AWS-Konto , zu dem die Rolle gehört, nicht im Element `NotPrincipal` enthalten wäre, könnte dies gleichermaßen zu einer Verweigerung des Zugriffs auf das AWS-Konto und allen Entitäten darin durch die Richtlinie führen. In einigen Fällen können Benutzer mit übernommener Rolle nicht mehr Berechtigungen als ihre übergeordnete Rolle haben, und Rollen können nicht mehr Berechtigungen als ihre übergeordnete AWS-Konto Rolle haben. Wenn also der Rolle oder dem Konto der Zugriff explizit verweigert wird, kann der Benutzer mit der angenommenen Rolle möglicherweise nicht auf die Ressource zugreifen.

Dieses Beispiel funktioniert wie beabsichtigt, wenn es Teil einer Richtlinienanweisung in einer ressourcenbasierten Richtlinie ist, die einer Ressource in einer anderen AWS-Konto (nicht 444455556666) zugeordnet ist. Dieses Beispiel allein gewährt dem Benutzer mit übernommener Rolle keinen Zugriff, es wird nur in der Liste der Prinzipale, die ausdrücklich verweigert wurden cross-account-audit-app, weggelassen. cross-account-audit-app Um cross-account-audit-app Zugriff auf die Ressource zu gewähren, muss eine andere Richtlinienanweisung den Zugriff über ausdrücklich zulassen. `"Effect": "Allow"`

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-
account-audit-app",
            "arn:aws:iam::444455556666:role/cross-account-read-only-role",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::Bucket_AccountAudit",
            "arn:aws:s3:::Bucket_AccountAudit/*"
        ]
    }]
}
```

Wenn Sie eine Sitzung mit übernommener Rolle in einem `NotPrincipal`-Element angeben, können Sie keinen Platzhalter (*) verwenden, der "alle Sitzungen" bedeutet. Auftraggeber müssen immer eine bestimmte Sitzung benennen.

IAM-JSON-Richtlinienelemente: Action

Das Element `Action` beschreibt die gewünschte oder gewünschten Aktionen, die zugelassen oder verweigert werden. Anweisungen müssen entweder ein `Action`- oder `NotAction`-Element enthalten. Jeder AWS Dienst hat seine eigenen Aktionen, die Aufgaben beschreiben, die Sie mit diesem Dienst ausführen können. Die Liste der Aktionen für Amazon S3 finden Sie beispielsweise unter Specifying Permissions in a Policy im Amazon Simple Storage Service User Guide, die Liste der Aktionen für Amazon EC2 finden Sie in der Amazon EC2 API Reference und die Liste der Aktionen für AWS Identity and Access Management finden Sie in der IAM API Reference. Die Liste

der Aktionen für andere Services finden Sie in der API-Referenz Dokumentation des jeweiligen Services.

Sie geben einen Wert über einen Namespace als Aktionspräfix (`iam`, `ec2` `sqs`, `sns`, `s3` usw.), gefolgt von dem Namen der Aktion, die erlaubt oder abgelehnt werden soll, an. Der Name muss mit einer Aktion übereinstimmen, die vom Service unterstützt wird. Der Präfix und Aktionsname wird nicht nach Groß- und Kleinschreibung unterschieden. Zum Beispiel ist `iam:ListAccessKeys` identisch zu `IAM:listaccesskeys`. In den folgenden Beispielen sind `Action`-Elemente für verschiedene Services aufgeführt.

Amazon SQS Aktion

```
"Action": "sqs:SendMessage"
```

Amazon EC2-Aktion

```
"Action": "ec2:StartInstances"
```

IAM-Aktion

```
"Action": "iam:ChangePassword"
```

Amazon S3-Aktionen

```
"Action": "s3:GetObject"
```

Sie können mehrere Werte für das Element `Action` definieren.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Sie können einen Platzhalter (*) verwenden, um Zugriff auf alle Aktionen zu gewähren, die das jeweilige Produkt anbietet. AWS Folgendes `Action`-Element beispielsweise findet für alle S3-Aktionen Anwendung.

```
"Action": "s3:*"
```

Sie können im Aktionsnamen auch Platzhalter (*) verwenden. Beispielsweise gilt folgendes `Action`-Element für alle IAM-Aktionen, die die Zeichenfolge `AccessKey` einschließlich `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys` und `UpdateAccessKey` enthalten.

```
"Action": "iam:*AccessKey*"
```

Bei einigen Services können Sie die verfügbaren Aktionen einschränken. Zum Beispiel stellt Amazon SQS nur einen Teil aller verfügbaren Amazon SQS-Aktionen bereit. In diesem Fall wird durch den *-Platzhalter keine vollständige Kontrolle über die Warteschlange gewährt, sondern nur diejenige Untergruppe von Aktionen zugelassen, die Sie geteilt haben. Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in Amazon S3 im Amazon Simple Storage Service Developer Guide.

IAM-JSON-Richtlinienelemente: NotAction

`NotAction` ist ein erweitertes Richtlinienelement, um eine explizite Übereinstimmung mit allen Aktionen herzustellen, mit Ausnahme der angegebenen Aktionsliste. Die Verwendung von `NotAction` kann zu einer kürzeren Richtlinie führen, da statt einer langen Liste von zugelassenen Aktionen lediglich einige wenige nicht zugelassenen Aktionen angegeben werden. Maßnahmen, `NotAction` die in angegeben sind, werden durch die `Allow` oder die `Deny` Wirkung in einer Grundsatzerklärung nicht beeinflusst. Dies bedeutet wiederum, dass alle nicht aufgelisteten einschlägigen Aktionen und Services zugelassen sind, wenn Sie die Anweisung `Allow` verwenden. Außerdem werden solche nicht aufgelisteten Aktionen und Services verweigert, wenn Sie die Anweisung `Deny` verwenden. Wenn Sie `NotAction` mit dem `Resource`-Element verwenden, schaffen Sie Raum für die Richtlinie. Auf diese Weise wird AWS bestimmt, welche Aktionen oder Dienste anwendbar sind. Weitere Informationen finden Sie in der folgenden Beispielrichtlinie.

NotAction mit Zulassen

Sie können das `NotAction` Element in einer Anweisung mit verwenden`"Effect": "Allow"`, um Zugriff auf alle Aktionen in einem AWS Dienst zu gewähren, mit Ausnahme der unter angegebenen Aktionen`NotAction`. Sie können sie mit dem `Resource`-Element verwenden, um Raum für die Richtlinie zu schaffen und die zugelassenen Aktionen auf die Aktionen zu beschränken, die für die angegebene Ressource ausgeführt werden können.

Das folgende Beispiel gestattet Benutzern den Zugriff auf die Amazon S3-Aktionen, die für jede S3-Ressource ausgeführt werden können, mit Ausnahme des Löschens eines Buckets. Dies gestattet den Benutzern nicht, die S3 API-Operation `ListAllMyBuckets` zu verwenden, weil für diese Aktion die Ressource "*" erforderlich ist. Diese Richtlinie lässt auch keine Aktionen in anderen Services zu, weil andere Service-Aktionen nicht auf die S3-Ressourcen anwendbar sind.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

Manchmal ist es sinnvoll, den Zugriff auf eine große Anzahl von Aktionen zu erlauben. Durch Verwendung des `NotAction`-Elements können Sie die Anweisung effektiv invertieren und so die Aktionsliste verkürzen. Da es beispielsweise so AWS viele Dienste gibt, möchten Sie vielleicht eine Richtlinie erstellen, die es dem Benutzer ermöglicht, alles zu tun, außer auf IAM-Aktionen zuzugreifen.

Das folgende Beispiel ermöglicht Benutzern den Zugriff auf jede Aktion in jedem AWS Dienst mit Ausnahme von IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Seien Sie vorsichtig bei der Verwendung der Elemente `NotAction` und `"Effect": "Allow"` in derselben Anweisung oder in einer anderen Anweisung innerhalb einer Richtlinie. `NotAction` stimmt mit allen Diensten und Aktionen überein, die nicht explizit aufgelistet oder auf die angegebene Ressource anwendbar sind und dazu führen könnten, dass Benutzer mehr Berechtigungen erhalten, als Sie beabsichtigt haben.

NotAction mit Deny

Sie können in einer Anweisung das Element `NotAction` mit `"Effect": "Deny"` verwenden, um den Zugriff auf alle aufgelisteten Ressourcen mit Ausnahme der im Element `NotAction` angegebenen Aktionen zu verweigern. Diese Kombination erteilt den aufgeführten Elementen keine Berechtigung, verweigert aber explizit den Zugriff auf die nicht aufgeführten Aktionen. Sie müssen unverändert die gewünschten Aktionen zulassen.

Das folgende Beispiel verweigert den Zugriff auf Nicht-IAM-Aktionen, wenn der Benutzer sich nicht mit MFA angemeldet hat. Hat sich der Benutzer mit MFA angemeldet, schlägt der `"Condition"`-Test fehl und die abschließende `"Deny"`-Anweisung hat keine Wirkung. Beachten Sie aber, dass dadurch kein Benutzerzugriff auf Aktionen gewährleistet ist. Es werden nur explizit alle anderen Aktionen als IAM-Aktionen verweigert.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

Eine Beispielrichtlinie, mit der der Zugriff auf Aktionen außerhalb bestimmter Regionen verweigert wird, mit Ausnahme von Aktionen aus bestimmten Services; siehe AWS: Verweigert den Zugriff auf AWS basierend auf der angeforderten Region.

IAM-JSON-Richtlinienelemente: Resource

Das Element `Resource` definiert das Objekt oder die Objekte, die von der Anweisung abgedeckt sind. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Geben Sie eine Ressource mit einem ARN an. Weitere Informationen zum Format von ARNs finden Sie unter IAM-ARNs.

Jeder Service verfügt über seine eigenen Ressourcen. Obwohl Sie immer zur Angabe einer Ressource einen ARN verwenden, hängen die ARN-Details für eine Ressource vom Service und der Ressource ab. Weitere Informationen zur Angabe einer Ressource finden Sie in der Dokumentation für den Service, für dessen Ressourcen Sie eine Anweisung definieren.

> **Note**
>
> Bei einigen Services können Sie keine Aktionen für einzelne Ressourcen angeben. Stattdessen gelten alle Aktionen, die Sie im Element `Action` oder `NotAction` auflisten, für sämtliche Ressourcen im jeweiligen Service. In diesem Fall nutzen Sie den Platzhalter `*` im `Resource`-Element.

Das folgende Beispiel bezieht sich auf eine bestimmte Amazon SQS-Warteschlange.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

Das folgende Beispiel bezieht sich auf den IAM-Benutzer mit dem Namen Bob in einem AWS-Konto.

> **Note**
>
> Im `Resource`-Element ist für den IAM-Benutzernamen die Groß- und Kleinschreibung zu beachten.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Verwenden von Platzhaltern in Ressourcen-ARNs

Sie können im ARN der Ressource Platzhalter verwenden. Sie können Platzhalterzeichen (* und?) innerhalb von ARN-Segmenten (die durch Doppelpunkte getrennten Teile) verwenden, um eine beliebige Kombination von Zeichen mit einem Asterisk (*) und einem beliebigen einzelnen Buchstaben mit einem Fragezeichen (?) darzustellen. Sie können mehrere * oder? -Zeichen in jedem Segment. Wenn der Platzhalter (*) das letzte Zeichen eines Ressourcen-ARN-Segments ist, kann er erweitert werden, sodass er über die Doppelpunktgrenzen hinaus übereinstimmt. Wir empfehlen die Verwendung von Platzhaltern (* und ?) innerhalb von ARN-Segmenten, die durch einen Doppelpunkt getrennt sind.

> **Note**
>
> Sie können im Servicesegment, das das AWS Produkt identifiziert, keinen Platzhalter verwenden. Weitere Informationen zu ARN-Segmenten finden Sie unter Amazon-Ressourcennamen (ARNs).

Das folgende Beispiel bezieht sich auf alle IAM-Benutzer, deren Pfad `/accounting` lautet.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

Das folgende Beispiel bezieht sich auf alle Elemente in einem spezifischen Amazon S3-Bucket.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

Das Sternchen (*) kann erweitert werden, um alles innerhalb eines Segments zu ersetzen, einschließlich Zeichen wie ein Schrägstrich (/), die andernfalls ein Trennzeichen innerhalb eines bestimmten Dienstnamespace zu sein scheinen. Betrachten Sie beispielsweise den folgenden Amazon S3 ARN, da dieselbe Platzhaltererweiterungslogik für alle Dienste gilt.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/test/*"
```

Die Platzhalter im ARN gelten für alle folgenden Objekte im Bucket, nicht nur für das erste aufgelistete Objekt.

```
DOC-EXAMPLE-BUCKET/1/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/3/object.jpg
DOC-EXAMPLE-BUCKET/1/2/3/test/4/object.jpg
DOC-EXAMPLE-BUCKET/1///test///object.jpg
DOC-EXAMPLE-BUCKET/1/test/.jpg
DOC-EXAMPLE-BUCKET//test/object.jpg
DOC-EXAMPLE-BUCKET/1/test/
```

Betrachten Sie die letzten beiden Objekte in der vorherigen Liste. Ein Amazon S3 Objektname kann gültig mit dem herkömmlichen Trennzeichen Schrägstrich (/) beginnen oder enden. Während „/“ als Trennzeichen fungiert, gibt es keine spezifische Bedeutung, wenn dieses Zeichen innerhalb eines Ressourcen-ARN verwendet wird. Es wird wie jedes andere gültige Zeichen behandelt. Der ARN würde nicht mit den folgenden Objekten übereinstimmen:

```
DOC-EXAMPLE-BUCKET/1-test/object.jpg
DOC-EXAMPLE-BUCKET/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test.jpg
```

Angabe mehrerer Aktionen oder Ressourcen

Sie können mehrere Ressourcen angeben. Das folgende Beispiel bezieht sich auf zwei DynamoDB-Tabellen.

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Verwenden von Richtlinienvariablen in Ressourcen-ARNs

Im `Resource`-Element können Sie JSON-Richtlinienvariablen in dem Teil des ARN verwenden, der die Ressource angibt (d. h. im abschließenden Teil des ARN). Beispielsweise können Sie den Schlüssel `{aws:username}` als Teil einer Ressourcen-ARN verwenden, damit der aktuelle Benutzername als Teil des Ressourcennamens aufgenommen wird. Das folgende Beispiel zeigt, wie

Sie den Schlüssel `{aws:username}` in einem `Resource`-Element verwenden können. Die Richtlinie gewährt Zugriff auf eine Amazon DynamoDB-Tabelle, die den Namen des aktuellen Benutzers enthält.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:account-id:table/${aws:username}"
  }
}
```

Weitere Informationen zu JSON-Richtlinienvariablen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

IAM-JSON-Richtlinienelemente: NotResource

`NotResource` ist ein erweitertes Richtlinienelement, das explizit jeder Ressource mit Ausnahme der angegebenen entspricht. Die Verwendung von `NotResource` kann zu einer kürzeren Richtlinie führen, da statt einer langen Liste von zulässigen Ressourcen lediglich einige wenige unzulässige Ressourcen angegeben werden. Dies ist besonders nützlich für Richtlinien, die innerhalb eines einzelnen AWS -Service gelten.

Stellen Sie sich beispielsweise vor, Sie haben eine Gruppe mit dem Namen `HRPayroll`. Mitglieder von `HRPayroll` sollten auf keine Amazon S3-Ressourcen außer dem Ordner `Payroll` im Bucket `HRBucket` zugreifen dürfen. Mit der folgenden Richtlinie wird der Zugriff auf alle nicht aufgelisteten Amazon S3-Ressourcen explizit verweigert. Beachten Sie jedoch, dass diese Richtlinie dem Benutzer keine Zugriffsberechtigung auf Ressourcen gewährt.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

Um einen Zugriff auf eine Ressource explizit zu verweigern, würden Sie normalerweise eine Richtlinie definieren, die ein `"Effect":"Deny"`- und `Resource`-Element enthält, in dem jeder Ordner einzeln aufgeführt ist. In diesem Fall müssen Sie allerdings jedes Mal, wenn Sie einen Ordner zu `HRBucket` oder eine Ressource zu Amazon S3 hinzufügen, auf die nicht zugegriffen werden darf, um deren Namen in der Liste der `Resource` zu ergänzen. Wenn Sie stattdessen ein `NotResource`-Element verwenden, wird den Benutzern der Zugriff auf neue Ordner automatisch verweigert, es sei denn, sie fügen die Ordnernamen zum `NotResource`-Element hinzu.

Bei der Verwendung von `NotResource` sollten Sie beachten, dass die in diesem Element angegebenen Ressourcen die einzigen nicht eingeschränkten Ressourcen sind. Dies wiederum schränkt alle Ressourcen ein, die für die Aktion gelten würden. Im obigen Beispiel wirkt sich die Richtlinie nur auf Amazon S3-Aktionen und daher nur auf Amazon S3-Ressourcen. Wenn die Aktion auch Amazon EC2-Aktionen enthält, verweigert die Richtlinie nicht den Zugriff auf EC2-Ressourcen. Informationen darüber, welche Aktionen in einem Dienst die Angabe des ARN einer Ressource ermöglichen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

NotResource mit anderen Elementen

Sie sollten die Elemente `"Effect": "Allow"`, `"Action": "*"` und `"NotResource": "arn:aws:s3:::HRBucket"` niemals zusammen verwenden. Diese Anweisung ist sehr gefährlich, da sie alle Aktionen AWS auf allen Ressourcen außer dem `HRBucket` S3-Bucket zulässt. Dies würde sogar dem Benutzer erlauben, selbst eine Richtlinie hinzuzufügen, die ihm Zugriff auf `HRBucket` gewährt. Tun Sie dies nicht.

Seien Sie vorsichtig bei der Verwendung der Elemente `NotResource` und `"Effect": "Allow"` in derselben Anweisung oder in einer anderen Anweisung innerhalb einer Richtlinie. `NotResource` erlaubt alle Dienste und Ressourcen, die nicht explizit aufgeführt sind, und könnte dazu führen, dass Benutzer mehr Berechtigungen erhalten, als Sie beabsichtigt haben. Beim Verwenden der Elemente `NotResource` und `"Effect": "Deny"` in derselben Anweisung werden nicht explizit aufgelistete Services und Ressourcen verweigert.

IAM-JSON-Richtlinienelemente: Condition

Mit dem Element `Condition` (oder dem `Condition`-Block) können Sie angeben, unter welchen Bedingungen eine Richtlinie wirksam ist. Das Element `Condition` ist optional. Im `Condition`-Element formulieren Sie Ausdrücke, in denen Sie Bedingungsoperatoren (gleich, kleiner als usw.) verwenden, um die Kontextschlüssel und -werte in der Richtlinie mit Schlüsseln und Werten im Anforderungskontext abzugleichen. Weitere Informationen zum Anforderungskontext finden Sie unter Anforderung.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

Der Kontextschlüssel, den Sie in einer Richtlinienbedingung angeben, kann ein globaler Bedingungskontextschlüssel oder ein servicespezifischer Kontextschlüssel sein. Kontextschlüssel für globale Bedingungen verfügen über das Präfix `aws:`. Servicespezifische Kontextschlüssel verfügen über das Präfix des Services. Mit Amazon EC2 können Sie beispielsweise mithilfe des `ec2:InstanceType`-Kontextschlüssels eine Bedingung schreiben, die für diesen Service eindeutig ist. Informationen zum Anzeigen servicespezifischer IAM-Kontextschlüssel mit dem Präfix `iam:` finden Sie unter IAM- und AWS STS Bedingungskontextschlüssel.

Bei Namen von Kontextschlüsseln wird die Groß-/Kleinschreibung nicht beachtet. Das Einbeziehen des `aws:SourceIP`-Kontextschlüssels ist beispielsweise gleichbedeutend mit dem Testen auf `AWS:SourceIp`. Die Berücksichtigung der Groß- und Kleinschreibung bei Werten von Kontextschlüsseln hängt vom verwendeten Bedingungsoperator ab. Die folgende Bedingung enthält beispielsweise den `StringEquals`-Operator, um sicherzustellen, dass nur von `johndoe` gestellte Anfragen übereinstimmen. Benutzern mit dem Namen `JohnDoe` wird der Zugriff verweigert.

```
"Condition" : { "StringEquals" : { "aws:username" : "johndoe" }}
```

Die folgende Bedingung verwendet den `StringEqualsIgnoreCase`-Operator, damit Benutzer mit dem Namen `johndoe` oder `JohnDoe` gefunden werden.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johndoe" }}
```

Einige Kontextschlüssel unterstützen Schlüssel-Wert-Paare, mit denen Sie einen Teil des Schlüsselnamens festlegen können. Beispiele hierfür sind der `aws:RequestTag/`*`tag-key`*Kontextschlüssel AWS KMS `kms:EncryptionContext:`*`encryption_context_key`*, der und der `ResourceTag/`*`tag-key`*Kontextschlüssel, die von mehreren Diensten unterstützt werden.

- Wenn Sie den `ResourceTag/`*`tag-key`*-Kontextschlüssel für einen Service wie etwa Amazon EC2 verwenden, müssen Sie einen Schlüsselnamen für den `tag-key` angeben.
- Bei den Schlüsselnamen muss die Groß- und Kleinschreibung nicht berücksichtigt werden. Dies bedeutet Folgendes: Wenn Sie `"aws:ResourceTag/TagKey1": "Value1"` im Bedingungselement Ihrer Richtlinie angeben, stimmt die Bedingung mit einem Ressourcen-Tag-Schlüssel mit dem Namen `TagKey1` oder `tagkey1` überein, aber nicht mit beiden.
- AWS Dienste, die diese Attribute unterstützen, ermöglichen es Ihnen möglicherweise, mehrere Schlüsselnamen zu erstellen, die sich nur durch Groß- und Kleinschreibung unterscheiden. Ein

Beispiel wäre hier das Tagging einer Amazon-EC2-Instance mit `ec2=test1` und `EC2=test2`. Wenn Sie eine Bedingung wie `"aws:ResourceTag/EC2": "test1"` verwenden, um den Zugriff auf diese Ressource zu erlauben, stimmt der Schlüsselname mit beiden Tags, jedoch nur mit einem Wert überein. Dies kann zu unerwarteten Bedingungsfehlern führen.

> **Important**
>
> Als bewährte Methode stellen Sie sicher, dass Mitglieder Ihres Kontos eine konsistente Namenskonvention beim Benennen von Schlüssel-Wert-Paar-Attributen verfolgen. Beispiele hierfür sind Tags oder AWS KMS -Verschlüsselungskontexte. Sie können dies erzwingen, indem Sie den `aws:TagKeys`Kontextschlüssel für das Tagging oder den `kms:EncryptionContextKeys`für den AWS KMS Verschlüsselungskontext verwenden.

- Eine Liste aller Bedingungsoperatoren und eine Beschreibung ihrer Funktionsweise finden Sie unter Bedingungsoperatoren.
- Sofern nicht anders angegeben, können alle Kontextschlüssel mehrere Werte haben. Eine Beschreibung zum Umgang mit Kontextschlüsseln mit mehreren Werten finden Sie unter Mehrwertige Kontextschlüssel.
- Eine Liste aller global verfügbaren Kontextschlüssel finden Sie unter AWS Kontextschlüssel für globale Bedingungen.
- Bedingungskontextschlüssel, die von jedem Dienst definiert werden, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste.

Der Anforderungskontext

Wenn ein Principal eine Anfrage an stellt AWS, fasst er AWS die Anforderungsinformationen in einem Anforderungskontext zusammen. Die Informationen werden verwendet, um die Anforderung auszuwerten und zu autorisieren. Sie können das `Condition`-Element einer JSON-Richtlinie verwenden, um bestimmte Kontextschlüssel anhand des Anforderungskontexts zu testen. Sie können beispielsweise eine Richtlinie erstellen, die den CurrentTime Kontextschlüssel aws: verwendet, um es einem Benutzer zu ermöglichen, Aktionen nur innerhalb eines bestimmten Zeitraums durchzuführen.

Wenn eine Anfrage eingereicht wird, AWS wertet sie jeden Kontextschlüssel in der Richtlinie aus und gibt den Wert true, false, not present und gelegentlich Null (eine leere Datenzeichenfolge) zurück. Ein Kontextschlüssel, der in der Anfrage nicht vorhanden ist, wird als Nichtübereinstimmung

betrachtet. Die folgende Richtlinie erlaubt beispielsweise das Entfernen Ihres eigenen Multi-Faktor-Authentifizierung (MFA)-Geräts, aber nur, wenn Sie sich in der letzten Stunde (3 600 Sekunden) per MFA angemeldet haben.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

Der Anforderungskontext kann die folgenden Werte zurückgeben:

- True – Wenn der Anforderer sich innerhalb der letzten Stunde per MFA angemeldet hat, gibt die Bedingung true zurück.
- False – Wenn sich der Anforderer vor mehr als einer Stunde per MFA angemeldet hat, gibt die Bedingung false zurück.
- Nicht vorhanden — Wenn der Anforderer eine Anfrage mit seinen IAM-Benutzerzugriffsschlüsseln in der AWS CLI AWS OR-API gestellt hat, ist der Schlüssel nicht vorhanden. In diesem Fall ist der Schlüssel nicht vorhanden und stimmt nicht überein.
- Null – Für Kontextschlüssel, die vom Benutzer definiert werden, z. B. die Übergabe von Tags in einer Anfrage, ist es möglich, eine leere Zeichenfolge einzuschließen. In diesem Fall ist der Wert im Anforderungskontext null. Ein Null-Wert kann in einigen Fällen "true" zurückgeben. Wenn Sie beispielsweise den mehrwertigen `ForAllValues`-Bedingungsoperator mit dem `aws:TagKeys`-Kontextschlüssel verwenden, können unerwartete Ergebnisse auftreten, wenn der Anforderungskontext null zurückgibt. Weitere Informationen finden Sie unter aws: TagKeys und. Mehrwertige Kontextschlüssel

Der Bedingungsblock

Das folgende Beispiel zeigt das grundlegende Format eines `Condition`-Elements:

```
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}
```

Ein Wert aus der Anfrage wird durch einen Kontextschlüssel dargestellt, in diesem Fall `s3:prefix`. Der Kontextschlüsselwert wird mit einem Wert verglichen, den Sie als Literalwert angeben, z. B. `janedoe/*`. Der vorzunehmende Vergleich wird vom Bedingungsoperator (hier `StringLike`) bestimmt. Sie können Bedingungen erstellen, die Zeichenketten, Datumsangaben, Zahlen und vieles mehr mit typischen booleschen Vergleichen wie "gleich", "größer als" und "kleiner als" vergleichen. Wenn Sie Zeichenfolgen-Operatoren oder ARN-Operatoren verwenden, können Sie auch eine Richtlinienvariable im Kontextschlüsselwert verwenden. Das folgende Beispiel enthält die Variable `aws:username`.

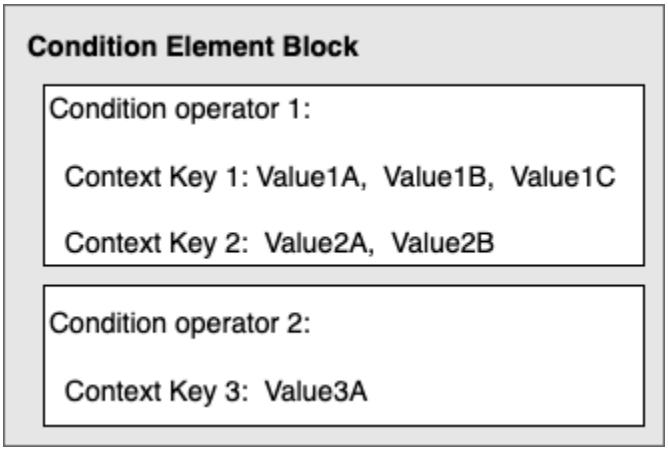
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

Unter bestimmten Umständen können Kontextschlüssel mehrere Werte enthalten. Eine Anforderung an Amazon DynamoDB könnte beispielsweise darin bestehen, mehrere Attribute einer Tabelle zurückzugeben oder zu aktualisieren. Eine Richtlinie für den Zugriff auf DynamoDB-Tabellen kann den `dynamodb:Attributes`-Kontextschlüssel enthalten, der alle in der Anfrage aufgeführten Attribute enthält. Sie können diese Attribute mit einer Liste zulässiger Attribute in einer Richtlinie mithilfe von Mengenoperatoren im Element `Condition` vergleichen. Weitere Informationen finden Sie unter Mehrwertige Kontextschlüssel.

Wenn die Richtlinie während einer Anfrage ausgewertet wird, wird der Schlüssel durch den entsprechenden Wert aus der Anfrage AWS ersetzt. (In diesem Beispiel AWS würden Datum und Uhrzeit der Anfrage verwendet.) Die Auswertung der Bedingung gibt "true" oder "false" zurück und das Ergebnis wird von der Richtlinie berücksichtigt, um die Anforderung zuzulassen oder zu verweigern.

Mehrere Werte in einer Bedingung

Ein `Condition`-Element kann mehrere Bedingungsoperatoren enthalten, und jeder Bedingungsoperator kann mehrere Kontext-Schlüssel-Wert-Paare enthalten. Dies wird in folgender Abbildung veranschaulicht.



Weitere Informationen finden Sie unter Mehrwertige Kontextschlüssel.

IAM-JSON-Richtlinienelemente: Bedingungsoperatoren

Verwenden Sie Bedingungsoperatoren im `Condition`-Element, um den Bedingungsschlüssel und -wert in der Richtlinie mit den Werten im Anforderungskontext abzugleichen. Weitere Informationen zum `Condition`-Element finden Sie unter IAM-JSON-Richtlinienelemente: Condition.

Der Bedingungsoperator, den Sie in einer Richtlinie verwenden können, hängt vom ausgewählten Bedingungsschlüssel ab. Sie können einen globalen Bedingungsschlüssel oder einen servicespezifischen Bedingungsschlüssel auswählen. Informationen dazu, welchen Bedingungsoperator Sie für einen globalen Bedingungsschlüssel verwenden können, finden Sie unter AWS Kontextschlüssel für globale Bedingungen. Informationen darüber, welchen Bedingungsoperator Sie für einen dienstspezifischen Bedingungsschlüssel verwenden können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste und wählen Sie den Service aus, den Sie anzeigen möchten.

> **Important**
>
> Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein und die Bedingung ist falsch. Wenn die Richtlinienbedingung erfordert, dass der Schlüssel nicht abgestimmt ist, wie `StringNotLike` oder `ArnNotLike`, und der richtige Schlüssel ist nicht vorhanden, ist die Bedingung wahr. Diese Logik gilt für alle Bedingungsoperatoren außer... IfExistsund Nullprüfung. Diese Operatoren testen, ob der Schlüssel im Anforderungskontext vorhanden ist (existiert).

Die Bedingungsoperatoren können in folgende Kategorien gruppiert werden:

- Zeichenfolge
- Numerischer Wert
- Datum und Uhrzeit
- Boolesch
- Binary
- IP-Adresse
- Amazon Ressourcenname (ARN) (nur für bestimmte Services verfügbar.)
- ... IfExists(prüft, ob der Schlüsselwert im Rahmen einer anderen Prüfung existiert)
- Null-Prüfung (überprüft als eigenständige Prüfung, ob der Schlüsselwert vorhanden ist)

Bedingungsoperatoren für Zeichenfolgen

Mit String-Bedingungsoperatoren können Sie `Condition`-Elemente erstellen, die den Zugriff basierend auf einen Vergleich eines Schlüssels mit einem Zeichenfolgewert einschränken.

Bedingungsoperator	Beschreibung
`StringEquals`	Exakte Übereinstimmung, Unterscheidung von Groß- und Kleinschreibung
`StringNotEquals`	Negierte Übereinstimmung
`StringEqualsIgnoreCase`	Exakte Übereinstimmung, keine Unterscheidung von Groß- und Kleinschreibung
`StringNotEqualsIgnoreCase`	Negierte Übereinstimmung, keine Unterscheidung von Groß- und Kleinschreibung
`StringLike`	Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung Die Werte können einen Mehrzeichen-Übereinstimmungs-Platzhalter (*) oder einen Einzelzeichen-Übereinstimmungs-Platzhalter (?) an einer beliebigen Stelle in der Zeichenfolge enthalten. Sie müssen Platzhalter angeben, um teilweise Zeichenfolgenübereinstimmungen zu erzielen.

Bedingungsoperator	Beschreibung
	Note Wenn ein Schlüssel mehrere Werte enthält, kann `StringLike` mit den Set-Operatoren `ForAllValues:StringLike` und `ForAnyValue:StringLike` ausgewertet werden. Weitere Informationen finden Sie unter Mehrwertige Kontextschlüssel.
`StringNotLike`	Negierte Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung Die Werte können einen Mehrzeichen-Übereinstimmungs-Platzhalter (*) oder einen Einzelzeichen-Übereinstimmungs-Platzhalter (?) an einer beliebigen Stelle in der Zeichenfolge enthalten.

Zum Beispiel enthält die folgende Anweisung ein `Condition`-Element, das den `aws:PrincipalTag`-Schlüssel verwendet, um anzugeben, dass der anfordernde Prinzipal mit der `iamuser-admin`-Aufgabenkategorie getaggt sein muss.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/job-category": "iamuser-admin"}}
  }
}
```

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein. In diesem Beispiel ist der Schlüssel `aws:PrincipalTag/job-category` im Anforderungskontext vorhanden, wenn der Auftraggeber einen IAM-Benutzer mit angehängten Tags verwendet. Er ist auch für einen Auftraggeber enthalten, der eine IAM-Rolle mit angehängten Tags oder Sitzungstags verwendet. Wenn ein Benutzer ohne das Tag versucht, einen Zugriffsschlüssel anzuzeigen oder zu bearbeiten, gibt die Bedingung `false` zurück und die Anforderung wird durch diese Anweisung implizit abgelehnt.

Sie können eine Richtlinienvariable mit dem `String`Bedingungsoperator verwenden.

Im folgenden Beispiel wird der `StringLike`-Bedingungsoperator verwendet, um einen String-Abgleich mit einer Richtlinienvariablen durchzuführen, um eine Richtlinie zu erstellen, die es einem IAM-Benutzer ermöglicht, die Amazon S3-Konsole zu verwenden, um sein eigenes "Heimatverzeichnis" in einem Amazon S3-Bucket zu verwalten. Die Richtlinie lässt die angegebenen Aktionen für ein S3-Bucket zu, solange `s3:prefix` einem angegebenen Muster entspricht.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

Ein Beispiel für eine Richtlinie, die zeigt, wie das `Condition` Element verwendet wird, um den Zugriff auf Ressourcen auf der Grundlage einer Anwendungs-ID und einer Benutzer-ID für den OIDC-

Verbund einzuschränken, finden Sie unter. Amazon S3: Ermöglicht Amazon Cognito-Benutzern den Zugriff auf Objekte in ihrem Bucket

Platzhalterabgleich

Operatoren für Zeichenfolgenbedingungen führen einen musterlosen Abgleich durch, der kein vordefiniertes Format erzwingt. Die Bedingungsoperatoren ARN und Datum sind eine Teilmenge von Zeichenfolgenoperatoren, die eine Struktur für den Bedingungsschlüsselwert erzwingen. Wenn Sie StringNotLike Operatoren StringLike oder für teilweise Zeichenfolgenübereinstimmungen eines ARN oder Datums verwenden, wird beim Abgleich ignoriert, welcher Teil der Struktur mit Platzhaltern versehen ist.

Die folgenden Bedingungen suchen beispielsweise mithilfe verschiedener Bedingungsoperatoren nach einer teilweisen Übereinstimmung eines ARN.

Wenn verwendet ArnLike wird, müssen die Partition, der Dienst, die Konto-ID, der Ressourcentyp und die teilweise Ressourcen-ID des ARN exakt mit dem ARN im Anforderungskontext übereinstimmen. Nur die Region und der Ressourcenpfad lassen einen teilweisen Abgleich zu.

```
"Condition": {"ArnLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

Wenn anstelle von verwendet StringLike wird ArnLike, ignoriert der Abgleich die ARN-Struktur und ermöglicht einen teilweisen Abgleich, unabhängig davon, welcher Teil mit Platzhaltern versehen wurde.

```
"Condition": {"StringLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

ARN	ArnLike	StringLike
arn:aws:cloudtrail:us-west-2:111122223333:trail/finance	Match	Match
arn:aws:cloudtrail:us-east-2:111122223333:trail/finance/archive	Match	Match
arn:aws:cloudtrail:us-east-2:444455556	Keine Übereinstimmung	Match

ARN	ArnLike	StringLike
666:user/111122223333:trail/finance		

Numerische Bedingungsoperatoren

Mit numerischen Bedingungsoperatoren können Sie `Condition`-Elemente erstellen, die den Zugriff basierend auf einen Vergleich eines Schlüssels mit einem Ganzzahl- oder Dezimalzahlwert einschränken.

Bedingungsoperator	Beschreibung
`NumericEquals`	Übereinstimmung
`NumericNotEquals`	Negierte Übereinstimmung
`NumericLessThan`	Übereinstimmung "Kleiner als"
`NumericLessThanEquals`	Übereinstimmung "Kleiner als oder gleich"
`NumericGreaterThan`	Übereinstimmung "Größer als"
`NumericGreaterThanEquals`	Übereinstimmung "Größer als oder gleich"

Die folgende Anweisung enthält beispielsweise ein `Condition`-Element, das den Bedingungsoperator `NumericLessThanEquals` mit dem Schlüssel `s3:max-keys` enthält, um anzugeben, dass der Anforderer bis zu 10 Objekte gleichzeitig im `example_bucket` aufnehmen kann.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
```

```
  }
}
```

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein. In diesem Beispiel ist der `s3:max-keys`-Schlüssel immer in der Anforderung vorhanden, wenn Sie die Operation `ListBucket` ausführen. Wenn diese Richtlinie alle Amazon S3-Operationen zulässt, sind nur die Operationen zulässig, die den `max-keys`-Kontextschlüssel mit einem Wert von kleiner oder gleich 10 enthalten.

Sie können eine Richtlinienvariable mit dem `Numeric`Bedingungsoperator verwenden.

Bedingungsoperatoren für Datum

Mit Date-Bedingungsoperatoren können Sie `Condition`-Elemente erstellen, die den Zugriff basierend auf einen Vergleich eines Schlüssels mit einem Datums-/Uhrzeitwert einschränken. Diese Bedingungsoperatoren werden mit dem Schlüssel `aws:CurrentTime` oder `aws:EpochTime` verwendet. Sie müssen die Datums-/Uhrzeitwerte unter Beachtung der W3C-Implementierungen der Datumsformate gemäß ISO 8601 oder in Epoch-Zeit (UNIX) angeben.

> **Note**
>
> Platzhalter für die Date-Bedingungsoperatoren sind unzulässig.

Bedingungsoperator	Beschreibung
`DateEquals`	Übereinstimmung mit einem bestimmten Datum
`DateNotEquals`	Negierte Übereinstimmung
`DateLessThan`	Übereinstimmung vor einem bestimmten Datum und einer bestimmten Uhrzeit
`DateLessThanEquals`	Übereinstimmung an oder vor einem bestimmten Datum und oder einer bestimmten Uhrzeit
`DateGreaterThan`	Übereinstimmung nach einem bestimmten Datum und einer bestimmten Uhrzeit

Bedingungsoperator	Beschreibung
`DateGreaterThanEquals`	Übereinstimmung an oder nach einem bestimmten Datum und einer bestimmten Uhrzeit

Beispielsweise enthält die folgende Anweisung ein `Condition`-Element, das den Bedingungsoperator `DateGreaterThan` mit dem `aws:TokenIssueTime`-Schlüssel verwendet. Diese Bedingung gibt an, dass die temporären Sicherheitsanmeldeinformationen, die für die Anforderung verwendet wurden, im Jahr 2020 ausgegeben wurden. Diese Richtlinie kann täglich programmgesteuert aktualisiert werden, um sicherzustellen, dass Kontomitglieder neue Anmeldeinformationen verwenden.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"DateGreaterThan": {"aws:TokenIssueTime": "2020-01-01T00:00:01Z"}}
  }
}
```

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein. Der `aws:TokenIssueTime`-Schlüssel ist im Anforderungskontext nur dann vorhanden, wenn der Auftraggeber temporäre Anmeldeinformationen für die Anforderung verwendet. Der Schlüssel ist weder in AWS API- noch in AWS CLI AWS SDK-Anfragen enthalten, die mithilfe von Zugriffsschlüsseln gestellt werden. Wenn in diesem Beispiel ein IAM-Benutzer versucht, einen Zugriffsschlüssel anzuzeigen oder zu bearbeiten, wird die Anforderung verweigert.

Sie können eine Richtlinienvariable mit dem `Date`Bedingungsoperator verwenden.

Boolesche Bedingungsoperatoren

Mit booleschen Bedingungsoperatoren können Sie `Condition`-Elemente erstellen, die den Zugriff basierend auf einen Vergleich eines Schlüssels mit "true" oder "false" einschränken.

Bedingungsoperator	Beschreibung
`Bool`	Boolesche Übereinstimmung

Zum Beispiel verwendet diese identitätsbasierte Richtlinie den Bedingungsoperator `Bool` mit dem Schlüssel `aws:SecureTransport`, um die Replikation von Objekten und Objekt-Tags in den Ziel-Bucket und dessen Inhalt zu verweigern, wenn die Anforderung nicht über SSL erfolgt.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die bestimmte Aktionen zulassen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein. Der `aws:SecureTransport`-Anforderungskontext gibt „true“ oder „false“ zurück.

Sie können eine Richtlinienvariable mit dem `Boolean`Bedingungsoperator verwenden.

Binäre Bedingungsoperatoren

Mit dem Bedingungsoperator `BinaryEquals` können Sie `Condition`-Elemente zum Prüfen von binären Schlüsselwerten erstellen. Er vergleicht den Wert des angegebenen Schlüssel Byte für Byte mit einer base-64-kodierten Darstellung des Binärwertes in der Richtlinie.

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein.

Sie können eine Richtlinienvariable mit dem `Binary`Bedingungsoperator verwenden.

Bedingungsoperatoren für IP-Adressen

Mit Bedingungsoperatoren für IP-Adressen können Sie `Condition`-Elemente erstellen, die den Zugriff anhand des Vergleichs eines Schlüssels mit einer IPv4- oder IPv6-Adresse bzw. einen IP-Adressbereich einschränken. Verwenden Sie sie mit dem Schlüssel `aws:SourceIp`. Der Wert muss im CIDR-Standardformat ausgedrückt werden (z. B. 203.0.113.0/24 oder 2001:DB8:1234:5678::/64). Wenn Sie eine IP-Adresse ohne das zugehörige Routing-Präfix zuweisen, verwendet IAM den Standard-Präfixwert `/32`.

Einige AWS Dienste unterstützen IPv6 und verwenden::, um einen Bereich von Nullen darzustellen. Um zu erfahren, ob ein Service IPv6 unterstützt, finden Sie in der Dokumentation des jeweiligen Service.

Bedingungsoperator	Beschreibung
`IpAddress`	Die angegebene IP-Adresse oder der angegebene IP-Bereich
`NotIpAddress`	Alle IP-Adressen mit Ausnahme der angegebenen IP-Adresse oder des angegebenen IP-Bereichs.

In der folgenden Anweisung wird beispielsweise der Bedingungsoperator `IpAddress` mit dem Schlüssel `aws:SourceIp` verwendet, um anzugeben, dass die Anforderung von einer IP-Adresse zwischen 203.0.113.0 und 203.0.113.255 kommen muss.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"IpAddress": {"aws:SourceIp": "203.0.113.0/24"}}
  }
}
```

Der Bedingungsschlüssel `aws:SourceIp` wird auf die IP-Adresse aufgelöst, von der die Anforderung kommt. Wenn die Anforderungen aus einer Amazon EC2-Instance kommen, gibt `aws:SourceIp` die öffentliche IP-Adresse der Instance zurück.

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein. Der `aws:SourceIp`-Schlüssel ist immer im Anforderungskontext vorhanden, außer wenn der Anforderer einen VPC-Endpunkt für die Anforderung verwendet. In diesem Fall gibt die Bedingung `false` zurück und die Anforderung wird durch diese Anweisung implizit abgelehnt.

Sie können eine Richtlinienvariable mit dem `IpAddress`Bedingungsoperator verwenden.

Im folgenden Beispiel wird dargestellt, wie Sie IPv4- und IPv6-Adressen verwenden können, um alle gültigen IP-Adressen in Ihrer Organisation abzudecken. Wir empfehlen, dass Sie die Richtlinien Ihrer Organisation zusätzlich zu den bereits vorhandenen IPv4-Adressbereichen um Ihre IPv6-Adressbereiche aktualisieren, um die Funktion Ihrer Richtlinien während der Umstellung auf IPv6 sicherzustellen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
```

```
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

Der Bedingungsschlüssel `aws:SourceIp` funktioniert in einer JSON-Richtlinie nur, wenn Sie die geprüfte API direkt als Benutzer aufrufen. Wenn Sie stattdessen einen Service verwenden, um den Zielservice in Ihrem Namen aufzurufen, erkennt der Zielservice die IP-Adresse des aufrufenden Services und nicht die IP-Adresse des verursachenden Benutzers. Dies kann beispielsweise passieren, wenn Sie Amazon EC2 aufrufen, AWS CloudFormation um Instances für Sie zu erstellen. Derzeit gibt es keine Möglichkeit, die ursprüngliche IP-Adresse über einen Aufrufservice an den Zielservice zur Bewertung in einer JSON-Richtlinie zu übermitteln. Verwenden Sie für diese Art von Service-API-Aufrufen nicht den Bedingungsschlüssel `aws:SourceIp`.

Bedingungsoperatoren für Amazon-Ressourcennamen (ARN)

Mit den Bedingungsoperatoren für Amazon-Ressourcennamen (ARN) können Sie `Condition`-Elemente erstellen, die den Zugriff basierend auf einen Vergleich eines Schlüssels mit einem ARN einschränken. Der ARN wird als Zeichenfolge interpretiert.

Bedingungsoperator	Beschreibung
`ArnEquals` , `ArnLike`	Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung des ARN Jede der sechs durch Doppelpunkt getrennten Komponenten der ARN wird separat überprüft und alle Komponenten können Mehrzeichen-Übereinstimmungs-Platzhalter (*) oder einen Einzelzeichen-Übereinstimmungs-Platzhalter (?) enthalten. Die `ArnEquals` - und `ArnLike`-Bedingungsoperatoren verhalten sich identisch.
`ArnNotEquals` , `ArnNotLike`	Negierte Übereinstimmung von ARN Die`ArnNotEquals` - und`ArnNotLike` -Bedingungsoperatoren verhalten sich identisch.

Sie können eine Richtlinienvariable mit dem `ARN`Bedingungsoperator verwenden.

Das folgende ressourcenbasierte Richtlinienbeispiel zeigt eine Richtlinie, die einer Amazon SQS-Warteschlange zugeordnet ist, an die Sie SNS-Nachrichten senden möchten. Es gibt Amazon SNS

die Erlaubnis, Nachrichten an die Warteschlange (oder Warteschlangen) Ihrer Wahl zu senden, aber nur, wenn der Service die Nachrichten im Namen eines bestimmten Amazon SNS-Themas (oder Themen) sendet. Geben Sie die Warteschlange im Feld `Resource` und das Amazon SNS-Thema als Wert für den Schlüssel `SourceArn` an.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "123456789012"},
    "Action": "SQS:SendMessage",
    "Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
    "Condition": {"ArnEquals": {"aws:SourceArn":
 "arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
  }
}
```

Wenn der Schlüssel, den Sie in einer Richtlinienbedingung angeben, im Anforderungskontext nicht vorhanden ist, stimmen die Werte nicht überein. Der `aws:SourceArn`-Schlüssel ist im Anforderungskontext nur dann vorhanden, wenn eine Ressource einen Service auslöst, um einen anderen Service im Namen des Ressourcenbesitzers aufzurufen. Wenn ein IAM-Benutzer versucht, diesen Vorgang direkt auszuführen, kehrt die Bedingung `false` zurück und die Anforderung wird implizit durch diese Anweisung abgelehnt.

... IfExists Bedingungsoperatoren

Sie können `IfExists` an das Ende jedes Bedingungsoperatornamens hinzufügen, abgesehen von der `Null`-Bedingung wie z. B. `StringLikeIfExists`. Damit geben Sie zum Ausdruck: "Wenn der Richtlinienschlüssel im Anforderungskontext vorhanden ist, wird der Schlüssel den Angaben in der Richtlinie entsprechend verarbeitet. Wenn der Schlüssel nicht vorhanden ist, wird das Bedingungselement als "true" ausgewertet. Andere Bedingungselemente in der Anweisung können weiterhin zu einer Nichtübereinstimmung führen, nicht jedoch ein fehlender Schlüssel bei einer Prüfung mit `...IfExists`. Wenn Sie ein `"Effect": "Deny"`-Element mit einem negierten Bedingungsoperator wie `StringNotEqualsIfExists` verwenden, wird die Anforderung auch dann abgelehnt, wenn das Tag fehlt.

Beispiel mit **`IfExists`**

Viele Bedingungsschlüssel beschreiben einen bestimmten Ressourcentyp und existieren nur beim Zugriff auf diesen Ressourcentyp. Diese Bedingungsschlüssel sind bei anderen Ressourcentypen

nicht vorhanden. Dies stellt kein Problem dar, wenn die Richtlinienanweisung nur für einen Ressourcentyp gültig ist. Es gibt jedoch Fälle, bei denen eine einzelne Anweisung auf mehrere Ressourcentypen zutrifft, wenn beispielsweise die Richtlinienanweisung auf Aktionen aus mehreren Services Bezug nimmt oder wenn eine bestimmte Aktion innerhalb eines Services auf mehrere verschiedene Ressourcentypen in demselben Service zugreift. In solchen Fällen kann ein nur für eine Ressource gültiger Bedingungsschlüssel in einer Richtlinienanweisung dazu führen, dass das Element `Condition` in der Richtlinienanweisung fehlschlägt, sodass der `"Effect"` der Anweisung nicht angewendet wird.

Betrachten Sie beispielsweise das folgende einfache Richtlinienbeispiel:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

Der Zweck der vorangegangenen Richtlinie besteht darin, den Benutzer zum Starten einer beliebigen Instance vom Typ `t1`, `t2` oder `m3` zu befähigen. Zum Starten einer Instance ist jedoch auch der Zugriff auf viele Ressourcen zusätzlich zu der Instance selbst erforderlich, wie z. B. Images, Schlüsselpaare, Sicherheitsgruppen usw. Die gesamte Anweisung wird gegen alle Ressourcen geprüft, die zum Starten der Instance erforderlich sind. Diese zusätzlichen Ressourcen verfügen nicht über den Bedingungsschlüssel `ec2:InstanceType`, sodass die Prüfung `StringLike` fehlschlägt und der Benutzer nicht zum Starten eines beliebigen Instance-Typs befähigt wird.

Um dieses Problem zu beheben, verwenden Sie stattdessen den Bedingungsoperator `StringLikeIfExists`. Auf diese Weise findet die Prüfung nur dann statt, wenn der Bedingungsschlüssel existiert. Sie können nachstehende Richtlinie folgendermaßen interpretieren: „Wenn die zu prüfende Ressource den Bedingungsschlüssel „`ec2:InstanceType`„ enthält, wird die Aktion nur zugelassen, wenn der Schlüsselwert mit `t1.`, `t2.` oder `m3.` beginnt.“ Wenn die zu prüfende Ressource nicht über diesen Bedingungsschlüssel verfügt, ist dies belanglos." Das

Sternchen (*) in den Bedingungsschlüsselwerten wird, wenn es mit dem `StringLikeIfExists`-Bedingungsoperator verwendet wird, als Platzhalter interpretiert, um teilweise übereinstimmende Zeichenfolgen zu erzielen. Die Anweisung `DescribeActions` enthält die Aktionen, die erforderlich sind, um die Instance in der Konsole anzeigen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RunInstance",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*",
            "Condition": {
                "StringLikeIfExists": {
                    "ec2:InstanceType": [
                        "t1.*",
                        "t2.*",
                        "m3.*"
             ]}}
        },
        {
            "Sid": "DescribeActions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeVpcs",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups"
            ],
            "Resource": "*"
        }]
}
```

Bedingungsoperator zur Prüfung der Existenz von Bedingungsoperatoren

Verwenden Sie einen Bedingungsoperator `Null`, um zu prüfen, ob ein Bedingungsschlüssel zum Zeitpunkt der Autorisierung abwesend ist. Verwenden Sie in der Richtlinienanweisung entweder `true` (der Schlüssel ist nicht vorhanden – der Wert beträgt null) oder `false` (der Schlüssel ist vorhanden und sein Wert ist ungleich null).

Sie können eine Richtlinienvariable mit dem `Null`Bedingungsoperator verwenden.

Beispielsweise können Sie mit diesem Bedingungsoperator bestimmen, ob ein Benutzer seine eigenen oder temporäre Anmeldeinformationen für den Vorgang benutzt. Wenn der Benutzer temporäre Anmeldeinformationen benutzt, ist der Schlüssel `aws:TokenIssueTime` vorhanden und hat einen Wert. Das folgende Beispiel zeigt eine Bedingung, die besagt, dass der Benutzer zur Nutzung der Amazon EC2-API keine temporären Anmeldeinformationen verwenden darf (der Schlüssel darf nicht vorhanden sein).

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"true"}}
  }
}
```

Bedingungen mit mehreren Kontextschlüsseln oder -werten

Sie können das `Condition`-Element einer Richtlinie verwenden, um mehrere Kontextschlüssel oder mehrere Werte für einen einzelnen Kontextschlüssel in einer Anfrage zu testen. Wenn Sie eine Anfrage an stellen AWS, entweder programmgesteuert oder über die AWS Management Console, enthält Ihre Anfrage Informationen über Ihren Principal, Ihren Betrieb, Ihre Tags und mehr. Mithilfe von Kontextschlüsseln können Sie die Werte der übereinstimmenden Kontextschlüssel in der Anfrage anhand der in der Richtlinienbedingung angegebenen Kontextschlüssel testen. Weitere Informationen zu den in einer Anforderung enthaltenen Informationen und Daten finden Sie unter Der Anforderungskontext.

Themen

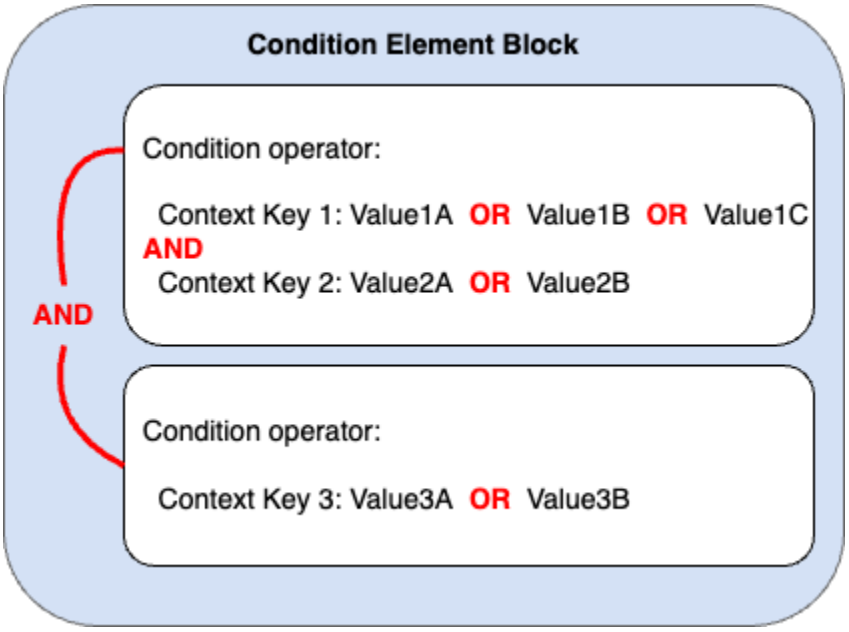
- Auswertungslogik für mehrere Kontextschlüssel oder -werte
- Auswertungslogik für negierte übereinstimmende Bedingungsoperatoren

Auswertungslogik für mehrere Kontextschlüssel oder -werte

Ein `Condition`-Element kann mehrere Bedingungsoperatoren enthalten, und jeder Bedingungsoperator kann mehrere Kontext-Schlüssel-Wert-Paare enthalten. Die meisten Kontextschlüssel unterstützen die Verwendung mehrerer Werte, sofern nicht anders angegeben.

- Wenn Ihre Richtlinienanweisung mehrere Bedingungsoperatoren enthält, werden die Bedingungsoperatoren anhand eines logischen AND ausgewertet.
- Wenn Ihre Richtlinienanweisung mehrere Bedingungsoperatoren enthält oder einem Bedingungsoperator mehrere Schlüssel angefügt sind, werden die Bedingungen anhand eines logischen AND ausgewertet.
- Wenn ein einzelner Bedingungsoperator mehrere Werte für einen Kontextschlüssel enthält, werden diese Werte anhand eines logischen OR ausgewertet.
- Wenn ein einzelner negierter übereinstimmender Bedingungsoperator mehrere Werte für einen Kontextschlüssel enthält, werden diese Werte anhand eines logischen NOR ausgewertet.

Alle Kontextschlüssel in einem Bedingungselementblock müssen zu „wahr“ aufgelöst werden, um den gewünschten `Allow`- oder `Deny`-Effekt aufzurufen. Das folgende Image veranschaulicht die Auswertungslogik für eine Bedingung mit mehreren Bedingungsoperatoren und Kontextschlüssel-Wert-Paaren.



Die folgende S3-Bucket-Richtlinie veranschaulicht beispielsweise, wie das vorherige Image in einer Richtlinie dargestellt wird. Der Bedingungsblock enthält die Bedingungsoperatoren `StringEquals` und `ArnLike` sowie die Kontextschlüssel `aws:PrincipalTag` und `aws:PrincipalArn`. Um den gewünschten `Allow`- oder `Deny`-Effekt aufzurufen, müssen alle Kontextschlüssel im Bedingungsblock zu „wahr“ aufgelöst werden. Der Benutzer, der die Anforderung stellt, muss beide Prinzipal-Tag-Schlüssel haben, Abteilung und Rolle, die einen der in der Richtlinie angegebenen Tag-Schlüsselwerte

enthalten. Außerdem muss der Prinzipal-ARN, der die Anfrage stellt, mit einem der in der Richtlinie angegebenen `aws:PrincipalArn`-Werte übereinstimmen, damit er als wahr ausgewertet werden kann.

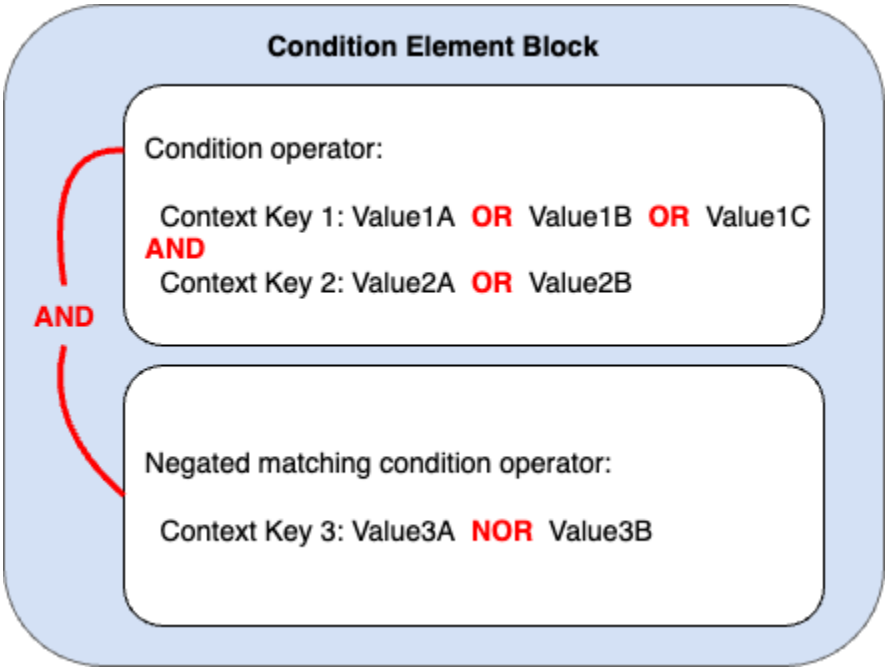
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Auswertungslogik für negierte übereinstimmende Bedingungsoperatoren

Einige Bedingungsoperatoren wie `StringNotEquals` oder `ArnNotLike` verwenden negierte Übereinstimmung, um die Kontext-Schlüssel-Wert-Paare in Ihrer Richtlinie mit den Kontext-Schlüssel-Wert-Paaren in einer Anfrage zu vergleichen. Wenn in einer Richtlinie mit negierten

übereinstimmenden Bedingungsoperatoren mehrere Werte für einen einzelnen Kontextschlüssel angegeben werden, funktionieren die effektiven Berechtigungen wie ein logisches NOR. Bei der negierten Übereinstimmung gibt ein logisches NOR oder NOT OR nur dann wahr zurück, wenn alle Werte als falsch ausgewertet werden.

Das folgende Image veranschaulicht die Auswertungslogik für eine Bedingung mit mehreren Bedingungsoperatoren und Kontextschlüssel-Wert-Paaren. Das Image enthält einen negierten übereinstimmenden Bedingungsoperator für Bedingungsschlüssel 3.



Die folgende S3-Bucket-Richtlinie veranschaulicht beispielsweise, wie das vorherige Image in einer Richtlinie dargestellt wird. Der Bedingungsblock enthält die Bedingungsoperatoren `StringEquals` und `ArnNotLike` sowie die Kontextschlüssel `aws:PrincipalTag` und `aws:PrincipalArn`. Um den gewünschten `Allow`- oder `Deny`-Effekt aufzurufen, müssen alle Kontextschlüssel im Bedingungsblock zu „wahr“ aufgelöst werden. Der Benutzer, der die Anforderung stellt, muss beide Prinzipal-Tag-Schlüssel haben, Abteilung und Rolle, die einen der in der Richtlinie angegebenen Tag-Schlüsselwerte enthalten. Da der `ArnNotLike`-Bedingungsoperator eine negierte Übereinstimmung verwendet, darf der Prinzipal-ARN des Benutzers, der die Anfrage stellt, mit keinem der in der Richtlinie angegebenen `aws:PrincipalArn`-Werte übereinstimmen, um als wahr ausgewertet zu werden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Einwertige im Vergleich zu mehrwertige Kontextschlüssel

Der Unterschied zwischen einwertigen und mehrwertigen Kontextschlüsseln hängt von der Anzahl der Werte im Anforderungskontext ab, nicht von der Anzahl der Werte in der Richtlinienbedingung.

- Einwertige Bedingungskontextschlüssel haben höchstens einen Wert im Anforderungskontext. Zum Beispiel können Sie Ressourcen in AWS mit Tags versehen. Ressourcen-Tags werden als Tag-Schlüssel-Wert-Paare gespeichert. Ein Ressourcen-Tag-Schlüssel kann einen einzelnen Tag-Wert haben. Daher handelt es sich bei the section called “ResourceTag” um einen einwertigen Kontextschlüssel. Verwenden Sie keinen Bedingungssatzoperator mit einem einwertigen Kontextschlüssel.

- Mehrwertige Bedingungskontextschlüssel können im Anforderungskontext mehrere Werte enthalten. Sie können beispielsweise Ressourcen in einer Anfrage taggen AWS und mehrere Tag-Schlüssel-Wert-Paare in eine Anfrage aufnehmen. Daher handelt es sich bei the section called "TagKeys" um einen mehrwertigen Kontextschlüssel. Mehrwertige Kontextschlüssel erfordern einen Bedingungssatz-Operator.

> **Important**
>
> Mehrwertige Kontextschlüssel erfordern einen Bedingungssatz-Operator. Verwenden Sie die Bedingungssatz-Operatoren `ForAllValues` oder `ForAnyValue` nicht mit einwertigen Kontextschlüsseln. Weitere Informationen zu Bedingungssatz-Operatoren finden Sie unter Mehrwertige Kontextschlüssel.

Die einwertigen und mehrwertigen Klassifizierungen sind in der Beschreibung jedes Bedingungskontextschlüssels als Werttyp im AWS Kontextschlüssel für globale Bedingungen-Thema enthalten. Die Referenz für Serviceautorisierung verwendet eine andere Werttypklassifizierung für mehrwertige Bedingungskontextschlüssel in folgendem Format: ein `ArrayOf`-Präfix, gefolgt vom Kategorietyp des Bedingungsoperators. Zum Beispiel `ArrayOfString` oder `ArrayOfARN`.

Beispielsweise kann eine Anfrage von höchstens einem VPC-Endpunkt stammen, weshalb es sich bei the section called "SourceVpce" um einen einwertigen Kontextschlüssel handelt. Da ein Service über mehr als einen Prinzipal verfügen kann, der zum Service gehört, handelt es sich bei aws:PrincipalService NamesList um einen mehrwertigen Kontextschlüssel.

Sie können jeden verfügbaren einwertigen Kontextschlüssel als Richtlinienvariable verwenden. Sie können keinen mehrwertigen Kontextschlüssel als Richtlinienvariable verwenden. Weitere Informationen zu Richtlinienvariablen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags.

Mehrwertige Kontextschlüssel erfordern die Bedingungssatzoperatoren `ForAllValues` oder `ForAnyValue`. Kontextschlüssel, die Schlüssel-Wert-Paare wie the section called "RequestTag" und the section called "ResourceTag" enthalten, können verwirren, da mehrere *`tag-key`*-Werte vorhanden sein können. Da aber jedes *`tag-key`* nur über einen Wert verfügen kann, sind `aws:RequestTag` und `aws:ResourceTag` jeweils einwertige Kontextschlüssel. Die Verwendung von Bedingungssatzoperatoren mit einwertigen Kontextschlüsseln kann zu übermäßig zulässigen Richtlinien führen.

Mehrwertige Kontextschlüssel

Um Ihren Bedingungskontextschlüssel mit einem Anforderungskontext-Schlüssel mit mehreren Werten zu vergleichen, müssen Sie die Satzoperatoren `ForAllValues` oder `ForAnyValue` verwenden. Diese Satzoperatoren werden verwendet, um zwei Sätze von Werten zu vergleichen, z. B. den Satz von Tags in einer Anfrage und den Satz von Tags in einer Richtlinienbedingung.

Die `ForAllValues`- und `ForAnyValue`-Qualifikatoren erweitern den Bedingungsoperator um die Funktion des Festlegens von Operationen, so dass Sie Anfragekontextschlüssel mit mehreren Werten gegen mehrere Bedingungskontextschlüsselwerte in einer Richtlinie testen können. Wenn Sie außerdem einen mehrwertigen Zeichenfolgen-Kontextschlüssel mit einem Platzhalter oder einer Variablen in Ihre Richtlinie einschließen, müssen Sie auch den `StringLike`-Bedingungsoperator verwenden. Mehrere Bedingungsschlüsselwerte müssen wie ein Array in Klammern eingeschlossen werden. z. B. `"Key2":["Value2A", "Value2B"]`.

- `ForAllValues` – Dieser Qualifikator prüft, ob der Wert jedes Elements des Anforderungssatzes eine Teilmenge des Bedingungskontextschlüsselsatzes ist. Die Bedingung gibt „wahr“ zurück, wenn jeder Kontextschlüsselwert in der Anfrage mit mindestens einem Kontextschlüsselwert in der Richtlinie übereinstimmt. Darüber hinaus wird „wahr“ zurückgegeben, wenn die Anfrage keine Kontextschlüssel enthält oder wenn der Kontextschlüsselwert in einen Nulldatensatz aufgelöst wird, z. B. eine leere Zeichenfolge. Um zu verhindern, dass fehlende Bedingungsschlüssel oder Bedingungsschlüssel mit leeren Werten als wahr ausgewertet werden, können Sie den Bedingungsoperator `Null` mit einem falschen Wert in Ihre Richtlinie einschließen. Dadurch wird geprüft, ob der Bedingungsschlüssel vorhanden ist und sein Wert nicht null ist.

  > **Important**
  >
  > Seien Sie vorsichtig, wenn Sie `ForAllValues` mit einem `Allow`-Effekt verwenden, da dies übermäßig zulässig sein kann, wenn das Vorhandensein fehlender Kontextschlüssel oder Kontextschlüssel mit leeren Werten im Anforderungskontext unerwartet ist. Sie können den `Null`-Bedingungsoperator mit einem falschen Wert in Ihre Richtlinie einschließen, um zu prüfen, ob der Kontextschlüssel vorhanden ist und sein Wert nicht null ist. Ein Beispiel finden Sie unter Zugriffssteuerung auf der Grundlage von Tag-Schlüsseln.

- `ForAnyValue` – Dieser Qualifikator prüft, ob mindestens ein Element des Satzes von Bedingungskontextschlüssel-Werten mit mindestens einem Element des Satzes von Bedingungskontextschlüssel-Werten in Ihrer Richtlinie übereinstimmt. Der Kontextschlüssel

gibt „wahr“ zurück, wenn einer der Kontextschlüsselwerte in der Anfrage mit einem der Kontextschlüsselwerte in der Richtlinie übereinstimmt. Wenn kein passender Kontextschlüssel vorhanden ist oder ein Nulldatensatz vorliegt, gibt die Bedingung „falsch“ zurück.

> **Note**
>
> Der Unterschied zwischen einwertigen und mehrwertigen Kontextschlüsseln hängt von der Anzahl der Werte im Anforderungskontext ab, nicht von der Anzahl der Werte in der Richtlinienbedingung.

Beispiele für Bedingungsrichtlinien

In IAM-Richtlinien können Sie mehrere Werte für einwertige und mehrwertige Kontextschlüssel für den Vergleich mit dem Anforderungskontext angeben. Die folgenden Richtlinienbeispiele veranschaulichen Richtlinienbedingungen mit mehreren Kontextschlüsseln und -werten.

> **Note**
>
> Wenn Sie eine Richtlinie zur Aufnahme in dieses Referenzhandbuch vorschlagen möchten, verwenden Sie die Schaltfläche Feedback unten auf der Seite. Beispiele für identitätsbasierte IAM-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien in IAM.

Beispiele für Bedingungsrichtlinien: Einwertige Kontextschlüssel

- Mehrere Bedingungsblöcke mit einwertigen Kontextschlüsseln. (Dieses Beispiel ansehen.)
- Ein Bedingungsblock mit mehreren einwertigen Kontextschlüsseln und -werten. (Dieses Beispiel ansehen.)

Beispiele für Bedingungsrichtlinien: Mehrwertige Kontextschlüssel

- Richtlinie mit Bedingungssatzoperator `ForAllValues` verweigern. (Dieses Beispiel ansehen.)
- Richtlinie mit Bedingungssatzoperator `ForAnyValue` verweigern. (Dieses Beispiel ansehen.)

Schlüsselbeispiele für mehrwertige Kontexte

Die folgenden Richtlinienbeispiele veranschaulichen, wie Richtlinienbedingungen mit mehrwertigen Kontextschlüsseln erstellt werden.

Beispiel: Richtlinie mit Bedingungssatzoperator ablehnen ForAllValues

Das folgende Beispiel für eine identitätsbasierte Richtlinie verweigert die Verwendung von IAM-Tagging-Aktionen, wenn bestimmte Tag-Schlüsselpräfixe in der Anfrage enthalten sind. Jeder Wert für den Kontextschlüssel `aws:TagKeys` enthält einen Platzhalter (*) für eine teilweise übereinstimmende Zeichenfolge. Die Richtlinie enthält den `ForAllValues`-Satz-Operator mit Kontextschlüssel `aws:TagKeys`, da der Anforderungskontextschlüssel mehrere Werte enthalten kann. Damit der Kontextschlüssel `aws:TagKeys` „wahr“ zurückgibt, muss jeder Wert in der Anfrage mit mindestens einem Wert in der Richtlinie übereinstimmen.

Der `ForAllValues`-Satz-Operator gibt auch „wahr“ zurück, wenn die Anfrage keine Kontextschlüssel enthält oder wenn der Kontextschlüsselwert in einen Nulldatensatz aufgelöst wird, beispielsweise eine leere Zeichenfolge. Um zu verhindern, dass fehlende Kontextschlüssel oder Kontextschlüssel mit leeren Werten als „wahr“ ausgewertet werden, schließen Sie den `Null`-Bedingungsoperator mit dem Wert `false` in Ihre Richtlinie ein, um zu überprüfen, ob der Kontextschlüssel in der Anforderung vorhanden ist und sein Wert nicht null ist.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die bestimmte Aktionen zulassen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
```

```
      ],
      "Condition": {
        "Null": {
          "aws:TagKeys": "false"
        },
        "ForAllValues:StringLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*",
            "key3*"
          ]
        }
      }
    }
  ]
}
```

Beispiel: Richtlinie mit Bedingungssatzoperator ablehnen ForAnyValue

Das folgende Beispiel für eine identitätsbasierte Richtlinie verweigert die Erstellung von Snapshots von EC2-Instance-Volumes, wenn Snapshots mit einem der in der Richtlinie angegebenen Tag-Schlüssel, `environment` oder `webserver`, gekennzeichnet sind. Die Richtlinie enthält den `ForAnyValue`-Satz-Operator mit Kontextschlüssel `aws:TagKeys`, da der Anforderungskontextschlüssel mehrere Werte enthalten kann. Wenn Ihre Tagging-Anfrage einen der in der Richtlinie angegebenen Tag-Schlüsselwerte enthält, gibt der `aws:TagKeys`-Kontextschlüssel den Wert wahr zurück und ruft den Effekt der Ablehnungsrichtlinie auf.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die bestimmte Aktionen zulassen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
```

```
            ],
            "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": ["environment", "webserver"]
                }
            }
        }
    ]
}
```

Beispiele für Richtlinien mit einwertigen Kontextschlüsseln

Die folgenden Richtlinienbeispiele veranschaulichen, wie Richtlinienbedingungen mit einwertigen Kontextschlüsseln erstellt werden.

Beispiel: Mehrere Bedingungsblöcke mit einwertigen Kontextschlüsseln

Wenn ein Bedingungsblock mehrere Bedingungen mit jeweils einem einzelnen Kontextschlüssel enthält, müssen alle Kontextschlüssel zu „wahr“ aufgelöst werden, damit der gewünschte `Allow`- oder `Deny`-Effekt aufgerufen wird. Wenn Sie negierte übereinstimmende Bedingungsoperatoren verwenden, wird die Auswertungslogik des Bedingungswerts umgedreht.

Das folgende Beispiel ermöglicht es Benutzern, EC2-Volumes zu erstellen und während der Volume-Erstellung Tags auf die Volumes anzuwenden. Der Anforderungskontext muss einen Wert für den Kontextschlüssel `aws:RequestTag/project` enthalten. Der Wert für den Kontextschlüssel `aws:ResourceTag/environment` kann ein beliebiger Wert außer „Produktion“ sein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:::volume/*",
      "Condition": {
```

```
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:region:account:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

Der Anforderungskontext muss einen Projekt-Tag-Wert enthalten und kann nicht für eine Produktionsressource zum Aufrufen des `Allow`-Effekts erstellt werden. Das folgende EC2-Volume wurde erfolgreich erstellt, da der Projektname `Feature3` mit einem `QA`-Ressourcen-Tag lautet.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Beispiel: Ein Bedingungsblock mit mehreren einwertigen Kontextschlüsseln und -werten

Wenn ein Bedingungsblock mehrere Kontextschlüssel enthält und jeder Kontextschlüssel über mehrere Werte verfügt, muss jeder Kontextschlüssel zu „wahr“ aufgelöst werden, damit mindestens ein Schlüsselwert für den gewünschten `Allow`- oder `Deny`-Effekt aufgerufen wird. Wenn Sie negierte übereinstimmende Bedingungsoperatoren verwenden, wird die Auswertungslogik des Wertes des Bedingungsschlüssels umgedreht.

Mit dem folgenden Beispiel können Benutzer Aufgaben in Clustern in Amazon Elastic Container Service starten und ausführen.

- Der Anforderungskontext muss `production` ODER `pre-prod` für den `aws:RequestTag/environment`-Kontextschlüssel UND enthalten.

- Der `ecs:cluster`-Kontextschlüssel stellt sicher, dass Aufgaben entweder auf den `default1` ODER `default2`-ARN-ECS-Clustern ausgeführt werden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

IAM-Richtlinienelemente: Variablen und Tags

Verwenden Sie AWS Identity and Access Management (IAM) -Richtlinienvariablen als Platzhalter, wenn Sie beim Schreiben der Richtlinie den genauen Wert einer Ressource oder eines Bedingungsschlüssels nicht kennen.

> **Note**
>
> Wenn eine Variable AWS nicht aufgelöst werden kann, kann dies dazu führen, dass die gesamte Anweisung ungültig ist. Wenn Sie beispielsweise die Variable `aws:TokenIssueTime` verwenden, wird diese nur dann in einen Wert aufgelöst, wenn der Anforderer für die Authentifizierung temporäre Anmeldeinformationen (eine IAM-Rolle) verwendet hat. Um zu verhindern, dass Variablen ungültige Anweisungen verursachen, verwenden Sie den... IfExists Bedingungsoperator.

Themen

Einführung

In IAM-Richtlinien ermöglichen Ihnen viele Aktionen, einen Namen für die speziellen Ressourcen bereitzustellen, für die Sie den Zugriff steuern möchten. Die folgende Richtlinie erlaubt Benutzern beispielsweise das Auflisten, Lesen und Schreiben von Objekten im S3-Bucket `DOC-EXAMPLE-BUCKET` für `marketing`-Projekte.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/marketing/*"]
    }
  ]
}
```

In manchen Fällen kennen Sie vielleicht den genauen Namen der Ressource nicht, wenn Sie die Richtlinie schreiben. Sie können die Richtlinie so verallgemeinern, dass sie für viele Benutzer funktioniert, ohne dass eine eindeutige Kopie der Richtlinie für jeden Benutzer erstellt werden muss. Anstatt für jeden Benutzer eine eigene Richtlinie zu erstellen, empfehlen wir Ihnen, eine einzelne Gruppenrichtlinie zu erstellen, die für jeden Benutzer in dieser Gruppe funktioniert.

Verwenden von Variablen in Richtlinien

Sie können dynamische Werte innerhalb von Richtlinien definieren, indem Sie Richtlinienvariablen verwenden, die Platzhalter in einer Richtlinie festlegen.

Variablen werden mit einem **$**-Präfix gefolgt von einem Paar geschweifter Klammern (**{ }**) markiert, die den Variablennamen des Werts aus der Anfrage enthalten.

Wenn die Richtlinie ausgewertet wird, werden die Richtlinienvariablen mit Werten aus den bedingten Kontextschlüsseln ersetzt, die in der Anforderung übermittelt wurden. Variablen können in identitätsbasierten Richtlinien, Ressourcenrichtlinien, Dienstverwaltungsrichtlinien, Sitzungsrichtlinien und VPC-Endpunkt-Richtlinien verwendet werden. Identitätsbasierte Richtlinien, die als Berechtigungsgrenzen verwendet werden, unterstützen auch Richtlinienvariablen.

Globale Bedingungskontextschlüssel können als Variablen in diensteübergreifenden AWS Anfragen verwendet werden. Servicespezifische Bedingungsschlüssel können bei der Interaktion mit AWS -Ressourcen auch als Variablen verwendet werden, sind jedoch nur verfügbar, wenn Anfragen an Ressourcen gestellt werden, die sie unterstützen. Eine Liste der für jeden AWS Dienst und jede Ressource verfügbaren Kontextschlüssel finden Sie in der Service Authorization Reference. Unter bestimmten Umständen können Sie globale Bedingungskontextschlüssel nicht mit einem Wert füllen. Weitere Informationen zu den einzelnen Schlüsseln finden Sie unter AWS Globale Bedingungskontextschlüssel.

Important

- Bei Schlüsselnamen wird nicht zwischen Groß- und Kleinschreibung unterschieden. Beispiel: `aws:CurrentTime` ist gleichbedeutend mit `AWS:currenttime`.
- Sie können jeden einwertigen Bedingungsschlüssel als Variable verwenden. Sie können einen mehrwertigen Bedingungsschlüssel nicht als Variable verwenden.

Das folgende Beispiel zeigt eine Richtlinie für eine IAM-Rolle oder einen IAM-Benutzer, die einen bestimmten Ressourcennamen durch eine Richtlinienvariable ersetzt. Sie können diese Richtlinie wiederverwenden, indem Sie den `aws:PrincipalTag`-Bedingungsschlüssel nutzen. Bei der Auswertung dieser Richtlinie lässt `${aws:PrincipalTag/team}` die Aktionen nur zu, wenn der Bucket-Name mit einem Teamnamen aus dem `team`-Prinzipal-Tag endet.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/team}/*"]
    }
  ]
}
```

Die Variable wird mithilfe des Präfix $, gefolgt von einem Paar geschweifter Klammern ({ }), gekennzeichnet. Innerhalb der ${ }-Zeichen können Sie den Namen des Wertes aus der Anforderung einfügen, den Sie in der Richtlinie verwenden möchten. Die Werte, die Sie verwenden können, werden weiter unten auf dieser Seite besprochen.

Einzelheiten zu diesem globalen Bedingungsschlüssel finden Sie unter PrincipalTagaws/tag-key in der Liste der globalen Bedingungsschlüssel.

> **Note**
>
> Zur Verwendung der Richtlinienvariablen müssen Sie das `Version`-Element in eine Anweisung einfügen und die Version muss auf eine Version gesetzt werden, die Richtlinienvariablen unterstützt. Variablen wurden in Version `2012-10-17` eingeführt. Frühere Versionen der Richtliniensprache unterstützen Richtlinienvariablen nicht. Wenn Sie das `Version`-Element nicht einfügen und nicht auf ein geeignetes Versionsdatum festlegen, werden Variablen wie `${aws:username}` als tatsächliche Zeichenfolgen in der Richtlinie behandelt.
>
> Das Richtlinienelement `Version` unterscheidet sich von einer Richtlinienversion. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Eine Richtlinienversion hingegen wird erstellt, wenn Sie in IAM eine kundenverwaltete Richtlinie bearbeiten. Die vorhandene Richtlinie wird von der geänderten Richtlinie nicht überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. Weitere Informationen zum Richtlinienelement `Version` finden Sie unter the section called “Version”. Weitere Informationen zu den Richtlinienversionen finden Sie unter the section called “Versioning von IAM-Richtlinien”.

Eine Richtlinie, die es einem Prinzipal ermöglicht, Objekte aus dem /David-Pfad eines S3-Buckets abzurufen, sieht wie folgt aus:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3::DOC-EXAMPLE-BUCKET/David/*"]
  }]
}
```

Wenn diese Richtlinie dem Benutzer `David` zugewiesen ist, erhält dieser Benutzer Objekte aus seinem eigenen S3-Bucket. Sie müssten jedoch für jeden Benutzer eine eigene Richtlinie erstellen, die den Namen des Benutzers enthält. Anschließend müssten Sie die einzelnen Richtlinien an die jeweiligen Benutzer anfügen.

Durch die Verwendung einer Richtlinienvariablen können Sie Richtlinien erstellen, die wiederverwendet werden können. Mit der folgenden Richtlinie kann ein Benutzer Objekte aus einem Amazon-S3-Bucket abrufen, wenn der Tag-Schlüsselwert für `aws:PrincipalTag` mit dem in der Anfrage übergebenen Tag-Schlüssel-Wert `owner` übereinstimmt.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Wenn Sie eine Richtlinienvariable anstelle eines solchen Benutzers verwenden, müssen Sie nicht für jeden einzelnen Benutzer eine eigene Richtlinie erstellen. Im folgenden Beispiel ist die Richtlinie einer IAM-Rolle zugeordnet, die von Produktmanagern mithilfe temporärer Sicherheitsanmeldeinformationen übernommen wird. Wenn ein Benutzer eine Anfrage zum Hinzufügen eines Amazon-S3-Objekts stellt, ersetzt IAM den `dept`-Tag-Wert aus der aktuellen Anfrage durch die `${aws:PrincipalTag}`-Variable und wertet die Richtlinie aus.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOnlyDeptS3Prefix",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/dept}/*"],
    }
  ]
}
```

Tags als Richtlinienvariablen

In einigen AWS Diensten können Sie Ressourcen, die von diesen Diensten erstellt wurden, Ihre eigenen benutzerdefinierten Attribute anhängen. Sie können beispielsweise Tags auf Amazon S3-Buckets oder IAM-Benutzer anwenden. Diese Tags sind Schlüssel-Wert-Paare. Sie definieren den Tag-Schlüsselnamen und den Wert, der diesem Schlüsselnamen zugeordnet ist. Beispielsweise können Sie ein Tag mit einem **`department`**-Schlüssel und einem **`Human Resources`**-Wert erstellen. Weitere Informationen zum Markieren von IAM-Entitäten finden Sie unter Markieren von IAM-Ressourcen. Informationen über das Markieren von Ressourcen, die von anderen AWS -Services erstellt wurden, finden Sie in der Dokumentation des jeweiligen Services. Informationen zur Verwendung des Tag-Editors finden Sie unter Arbeiten mit dem Tag-Editor im AWS Management Console -Leitfaden.

Sie können IAM-Ressourcen markieren, um das Entdecken, Organisieren und Nachverfolgen Ihrer IAM-Ressourcen zu vereinfachen. Sie können IAM-Identitäten auch markieren, um den Zugriff auf Ressourcen oder auf die Markierung zu kontrollieren. Weitere Informationen über die Verwendung von Tags zur Zugriffskontrolle finden Sie unter Steuerung des Zugriffs auf und für IAM-Benutzer und IAM-Rollen mithilfe von Tags.

Wo Richtlinienvariablen verwendet werden können

Sie können Richtlinienvariablen im `Resource`-Element und in Zeichenfolgenvergleichen im `Condition`-Element verwenden.

Ressourcen-Element

Sie können eine Richtlinienvariable im `Resource`-Element verwenden, jedoch nur im Ressourcenbereich des ARN. Dieser Teil des ARN erscheint nach dem 5. Doppelpunkt (:). Sie können keine Variable verwenden, um Teile des ARN vor dem 5. Doppelpunkt zu ersetzen, z. B. den Service oder das Konto. Weitere Informationen zum ARN-Format finden Sie unter IAM-ARNs.

Um einen Teil eines ARN durch einen Tag-Wert zu ersetzen, umgeben Sie das Präfix und den Schlüsselnamen mit `${ }`. Das folgende Resource-Element bezieht sich beispielsweise nur auf einen Bucket, dessen Name mit dem Wert im Abteilungs-Tag des anfordernden Benutzers übereinstimmt.

```
"Resource": ["arn:aws::s3:::bucket/${aws:PrincipalTag/department}"]
```

Viele AWS Ressourcen verwenden ARNs, die einen vom Benutzer erstellten Namen enthalten. Die folgende IAM-Richtlinie stellt sicher, dass nur vorgesehene Benutzer mit übereinstimmenden access-project-, access-project- und access-environment-Tag-Werten ihre Ressourcen ändern

können. Darüber hinaus können sie durch die Verwendung von *-Platzhalterübereinstimmungen benutzerdefinierte Suffixe für Ressourcennamen zulassen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
  ]
}
```

Condition-Element

Sie können eine Richtlinienvariable für `Condition`-Werte in jeder Bedingung verwenden, die die String-Operatoren oder die ARN-Operatoren beinhaltet. String-Operatoren beinhalten `StringEquals`, `StringLike` und `StringNotLike`. ARN-Operatoren beinhalten `ArnEquals` und `ArnLike`. Sie können eine Richtlinienvariable nicht mit anderen Operatoren wie `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address` oder `Null` verwenden. Weitere Hinweise zu Bedingungsoperatoren finden Sie unter IAM-JSON-Richtlinienelemente: Bedingungsoperatoren.

Beim Referenzieren eines Tags in einem `Condition`-Elementausdruck verwenden Sie das entsprechende Präfix und den Schlüsselnamen als Bedingungsschlüssel. Verwenden Sie dann den Wert, den Sie testen möchten, im Bedingungswert.

Das folgende Richtlinienbeispiel ermöglicht Benutzern beispielsweise vollständigen Zugriff, jedoch nur, wenn dem Benutzer das Tag `costCenter` zugeordnet ist. Das Tag muss auch einen Wert von entweder `12345` oder `67890` haben. Wenn das Tag keinen Wert oder einen anderen Wert hat, schlägt die Anforderung fehl.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Richtlinienvariablen ohne Wert

Wenn Richtlinienvariablen auf einen Bedingungskontextschlüssel verweisen, der keinen Wert hat oder in einem Autorisierungskontext für eine Anforderung nicht vorhanden ist, ist der Wert effektiv Null. Es gibt keinen gleichen oder ähnlichen Wert. Bedingungskontextschlüssel sind im Autorisierungskontext möglicherweise nicht vorhanden, wenn:

- Sie servicespezifische Bedingungskontextschlüssel in Anfragen an Ressourcen verwenden, die diesen Bedingungsschlüssel nicht unterstützen.
- Tags für IAM-Prinzipale, Sitzungen, Ressourcen oder Anfragen nicht vorhanden sind.
- Andere Umstände, wie sie für jeden globalen Bedingungskontextschlüssel in AWS Kontextschlüssel für globale Bedingungen aufgeführt sind.

Wenn Sie eine Variable ohne Wert im Bedingungselement einer IAM-Richtlinie verwenden, und ein IAM-JSON-Richtlinienelemente: Bedingungsoperatoren wie `StringEquals` oder `StringLike` nicht übereinstimmen, wird die Richtlinienanweisung nicht wirksam.

Invertierte Bedingungsoperatoren wie `StringNotEquals` oder `StringNotLike` entsprechen einem Nullwert oder stimmen mit ihm überein, da der Wert des Bedingungsschlüssels, anhand dessen sie testen, nicht dem tatsächlichen Nullwert entspricht oder diesem entspricht.

Im folgenden Beispiel muss `aws:principaltag/Team` gleich `s3:ExistingObjectTag/Team` sein, um den Zugriff zu gewähren. Der Zugriff wird ausdrücklich verweigert, wenn `aws:principaltag/Team` nicht festgelegt ist. Wenn eine Variable, die im Autorisierungskontext keinen Wert hat, als Teil des `Resource`- oder `NotResource`-Elements einer Richtlinie verwendet wird, entspricht die Ressource, die eine Richtlinienvariable ohne Wert enthält, keiner Ressource.

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::/example-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
    }
  ]
}
```

Anforderungsinformationen, die Sie für Richtlinienvariablen verwenden können

Sie können das `Condition`-Element einer JSON-Richtlinie verwenden, um Schlüssel im Anforderungskontext mit Schlüsselwerten zu vergleichen, die Sie in Ihrer Richtlinie angeben. Wenn Sie eine Richtlinienvariable verwenden, wird die Variable in Ihrer Richtlinie durch einen Wert aus dem Anforderungskontextschlüssel AWS ersetzt.

Auftraggeber-Schlüsselwerte

Die Werte für `aws:username`, `aws:userid` und `aws:PrincipalType` hängen davon ab, welche Art von Auftraggeber die Anforderung ausgelöst hat. Die Anforderung kann beispielsweise über die Anmeldeinformationen eines IAM-Benutzers, einer IAM-Rolle oder des Root-Benutzer des AWS-Kontos s erfolgen. In der folgenden sind die Werte für diese Schlüssel für verschiedene Auftraggebertypen aufgelistet.

- Root-Benutzer des AWS-Kontos
  - `aws:username`: (Nicht vorhanden)
  - `aws:userid`: AWS-Konto ID
  - `aws:PrincipalType`: `Account`
- IAM-Benutzer
  - `aws:username`: *`IAM-Benutzername`*
  - `aws:userid`: Eindeutige ID
  - `aws:PrincipalType`: `User`

- Verbundbenutzer
  - `aws:username`: (Nicht vorhanden)
  - `aws:userid`: *`Konto`*:*`Vom-Aufrufer-angegebener-Name`*
  - `aws:PrincipalType`: `FederatedUser`
- Web- und SAML-Verbundbenutzer

> **Note**
>
> Informationen zu Richtlinienschlüsseln, die verfügbar sind, wenn Sie den OIDC-Verbund verwenden, finden Sie unter. ???

  - `aws:username`: (Nicht vorhanden)
  - `aws:userid`: (Nicht vorhanden)
  - `aws:PrincipalType`: `AssumedRole`
- Angenommene Rolle
  - `aws:username`: (Nicht vorhanden)
  - `aws:userid`: *`Rollen-ID`*:*`Vom-Aufrufer-angegebener-Rollenname`*
  - `aws:PrincipalType`: `Assumed role`
- Eine einer Amazon EC2-Instance zugeordnete Rolle
  - `aws:username`: (Nicht vorhanden)
  - `aws:userid`: *`Rollen-ID`*:*`EC2-Instance-ID`*
  - `aws:PrincipalType`: `Assumed role`
- Anonymer Anrufer (Nur Amazon SQS Amazon SNS und Amazon S3)
  - `aws:username`: (Nicht vorhanden)
  - `aws:userid`: (Nicht vorhanden)
  - `aws:PrincipalType`: `Anonymous`

Beachten Sie für die Elemente in dieser Liste Folgendes:

- Nicht vorhanden bedeutet, dass der Wert nicht in den aktuellen Anforderungsinformationen vorhanden ist und jeder Versuch, ihn zuzuordnen, fehlschlägt und dafür sorgt, dass die Anforderung abgelehnt wird.

- *Rollen-ID* ist ein eindeutiger Bezeichner, der jeder Rolle bei der Erstellung zugeordnet wird. Sie können die Rollen-ID mit dem AWS CLI folgenden Befehl anzeigen: `aws iam get-role --role-name` *rolename*
- *Vom-Aufrufer-angegebener-Name* und *Vom-Aufrufer-angegebener-Rollenname* sind Namen, die vom aufrufenden Prozess (wie etwa einer Anwendung oder einem Service) übergeben werden, wenn er einen Aufruf tätigt, um temporäre Anmeldeinformationen zu erhalten.
- *EC2-Instance-ID* ist ein Wert, der der Instance zugeordnet wird, wenn sie gestartet und auf der Seite Instances der Amazon EC2-Konsole angezeigt wird. Sie können die Instanz-ID auch anzeigen, indem Sie den AWS CLI folgenden Befehl ausführen: `aws ec2 describe-instances`

In Anforderungen für Verbundbenutzer verfügbare Informationen

Verbundbenutzer sind Benutzer, die über ein anderes System als IAM authentifiziert werden. Beispielsweise könnte ein Unternehmen eine Anwendung für den internen Gebrauch haben, die Anrufe tätigt an AWS. Es könnte unpraktisch sein, jedem Unternehmensbenutzer, der die Anwendung verwendet, eine IAM-Identität zu geben. Das Unternehmen kann stattdessen eine Proxy-Anwendung (Middle-Tier) verwenden, die eine einzige IAM-Identität aufweist, oder das Unternehmen könnte einen SAML-Identitätsanbieter nutzen. Die Proxy-Anwendung oder der SAML-Identitätsanbieter authentifiziert einzelne Benutzer, die das Unternehmensnetzwerk verwenden. Eine Proxy-Anwendung kann dann ihre IAM-Identität verwenden, um temporäre Sicherheitsanmeldeinformationen für einzelne Benutzer abzurufen. Ein SAML-IdP kann tatsächlich Identitätsinformationen gegen AWS temporäre Sicherheitsanmeldeinformationen austauschen. Die temporären Anmeldeinformationen können dann für den Zugriff AWS auf Ressourcen verwendet werden.

Ebenso können Sie eine Anwendung für ein mobiles Gerät erstellen, in dem die Anwendung auf AWS -Ressourcen zugreifen muss. In diesem Fall können Sie den OIDC-Verbund verwenden, bei dem die App den Benutzer mithilfe eines bekannten Identitätsanbieters wie Login with Amazon, Amazon Cognito, Facebook oder Google authentifiziert. Die Anwendung kann dann die Authentifizierungsinformationen des Benutzers von diesen Anbietern verwenden, um temporäre Sicherheitsanmeldeinformationen für den Zugriff auf AWS -Ressourcen abzurufen.

Die empfohlene Methode zur Verwendung des OIDC-Verbunds besteht darin, Amazon Cognito und die AWS mobilen SDKs zu nutzen. Weitere Informationen finden Sie hier:

- Amazon-Cognito-Benutzerhandbuch
- Gängige Szenarien für temporäre Anmeldeinformationen

Sonderzeichen

Es gibt einige spezielle vordefinierte Richtlinienvariablen, die feste Werte haben. Diese bieten Ihnen die Möglichkeit, Zeichen darzustellen, die sonst eine besondere Bedeutung haben. Wenn diese Sonderzeichen Teil der Zeichenfolge sind, die Sie versuchen zusammenzubringen, und Sie sie einfach einfügen, würden sie fehlinterpretiert werden. Beispielsweise würde das Einfügen eines Sternchens (*) in die Zeichenfolge als Platzhalter interpretiert werden, der für andere Zeichen steht, und nicht tatsächlich als *. In solchen Fällen können Sie die folgenden vordefinierten Richtlinienvariablen verwenden:

- ${*} – zu verwenden, wenn Sie ein * (Sternchen) benötigen
- ${?} – zu verwenden, wenn Sie ein ? (Fragezeichen) benötigen
- ${$} – zu verwenden, wenn Sie ein $ (Dollarzeichen) benötigen

Diese vordefinierten Richtlinienvariablen können in einer beliebigen Zeichenfolge verwendet werden, in der Sie reguläre Richtlinien nutzen können.

Festlegen von Standardwerten

Um einer Variablen einen Standardwert hinzuzufügen, umgeben Sie den Standardwert mit einfachen Anführungszeichen (`' '`), und trennen Sie den Variablentext und den Standardwert durch Komma und Leerzeichen (`, `) enthalten.

Zum Beispiel, wenn ein Prinzipal mit `team=yellow` gekennzeichnet ist, können sie auf `ExampleCorp's`-Amazon-S3-Bucket namens `DOC-EXAMPLE-BUCKET-yellow` zugreifen. Eine Richtlinie mit dieser Ressource ermöglicht es Teammitgliedern, auf ihren Team-Bucket zuzugreifen, nicht jedoch auf die anderer Teams. Für Benutzer ohne Team-Tags wird der Standardwert von `company-wide` für den Bucket-Namen eingestellt. Diese Benutzer können nur auf `DOC-EXAMPLE-BUCKET-company-wide`Bucket zugreifen, wo umfassende Informationen anzeigt werden können, z. B. Anweisungen für den Beitritt zu einem Team.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Weitere Informationen

Weitere Informationen zu Richtlinien finden Sie in den folgenden Themen:

- Berechtigungen und Richtlinien in IAM
- Beispiele für identitätsbasierte Richtlinien in IAM

- IAM-JSON-Richtlinienelementreferenz
- Auswertungslogik für Richtlinien
- OIDC-Föderation

IAM-JSON-Richtlinienelemente: Unterstützte Datentypen

In diesem Abschnitt werden die Datentypen aufgeführt, die beim Festlegen von Werten in den JSON-Richtlinien unterstützt werden. Die Richtliniensprache unterstützt nicht alle Datentypen für jedes Richtlinienelement. Weitere Informationen zum jeweiligen Element finden Sie in den vorherigen Abschnitten.

- Zeichenfolgen
- Zahlen (Ganzzahlen oder Gleitkommazahlen)
- Boolesch
- Null
- Listen
- Zuordnungen
- Strukturen (dies sind lediglich verschachtelte Zuordnungen)

In der folgenden Tabelle wird jeder Datentyp der entsprechenden Serialisierung zugeordnet. Beachten Sie, dass alle Richtlinien UTF-8-kodiert sein müssen. Wenn Sie weitere Informationen zu den JSON-Datentypen erhalten möchten, rufen Sie die Website RFC 4627 auf.

Typ	JSON
String	String
Ganzzahl	Zahl
Gleitkommazahl	Zahl
Boolesch	wahr falsch
Null	Null
Datum	Zeichenfolge, die dem W3C-Profil von ISO 8601 entspricht

Typ	JSON
IpAddress	Zeichenfolge, die RFC 4632 erfüllt
Auflisten	Array
Object	Object

Auswertungslogik für Richtlinien

Wenn ein Principal versucht AWS Management Console, die AWS API oder den AWS CLI zu verwenden, sendet dieser Principal eine Anfrage. AWS Wenn ein AWS Dienst die Anfrage empfängt, AWS führt er mehrere Schritte durch, um zu bestimmen, ob die Anfrage zugelassen oder abgelehnt werden soll.

1. Authentifizierung — authentifiziert AWS zunächst den Prinzipal, der die Anfrage stellt, falls erforderlich. Dieser Schritt ist für einige wenige Services, wie z. B. Amazon S3, die einige Anforderungen von anonymen Benutzern erlauben, nicht notwendig.
2. Verarbeitung des Anforderungskontexts— AWS verarbeitet die in der Anfrage gesammelten Informationen, um festzustellen, welche Richtlinien für die Anfrage gelten.
3. Auswerten von Richtlinien in einem einzelnen Konto— AWS bewertet alle Richtlinientypen, was sich auf die Reihenfolge auswirkt, in der die Richtlinien bewertet werden.
4. Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird— verarbeitet AWS dann die Richtlinien anhand des Anforderungskontextes, um festzustellen, ob die Anfrage zugelassen oder abgelehnt wird.

Verarbeitung des Anforderungskontexts

AWS verarbeitet die Anfrage, um die folgenden Informationen in einem Anfragekontext zu sammeln:

- Aktionen (oder Operationen) – Die Aktionen oder Operationen, die der Auftraggeber durchführen möchte.
- Ressourcen — Das AWS Ressourcenobjekt, für das die Aktionen oder Operationen ausgeführt werden.

- Auftraggeber – Der Benutzer, die Rolle, der Verbundbenutzer oder die Anwendung, der bzw. die die Anfrage gesendet hat. Informationen über den Auftraggeber beinhalten die Richtlinien, die diesem Auftraggeber zugeordnet sind.
- Umgebungsdaten – Informationen über die IP-Adresse, den Benutzeragent, den SSL-Status oder die Tageszeit.
- Ressourcendaten – Daten zu den angeforderten Ressourcen. Dies sind zum Beispiel Informationen wie ein DynamoDB-Tabellenname oder Tag auf einer Amazon EC2-Instance.

AWS verwendet diese Informationen dann, um Richtlinien zu finden, die für den Anforderungskontext gelten.

Auswerten von Richtlinien in einem einzelnen Konto

Wie Richtlinien AWS bewertet werden, hängt von den Arten von Richtlinien ab, die für den Anforderungskontext gelten. Die folgenden Richtlinientypen, aufgelistet in der Reihenfolge ihrer Häufigkeit, stehen für die Verwendung in einem einzelnen AWS-Konto zur Verfügung. Weitere Informationen zu diesen Richtlinientypen finden Sie unter Berechtigungen und Richtlinien in IAM. Informationen darüber, wie Richtlinien für den kontoübergreifenden Zugriff AWS bewertet werden, finden Sie unter. Logik für die kontenübergreifende Richtlinienauswertung

1. Identitätsbasierte Richtlinien – Identitätsbasierte Richtlinien werden einer IAM-Identität (Benutzer, Benutzergruppe oder Rolle) angefügt und gewähren IAM-Entitäten (Benutzer und Rollen) Berechtigungen. Wenn für eine Anfrage nur identitätsbasierte Richtlinien gelten, AWS überprüft es alle Richtlinien auf mindestens eine. `Allow`
2. Ressourcenbasierte Richtlinien - Ressourcenbasierte Richtlinien erteilen dem angegebenen Auftraggeber (Konto, Benutzer, Rolle und Prizipalsitzungen wie Rollensitzungen und IAM-Verbundbenutzer) Berechtigungen, die als Prinzipal angegeben werden. Die Berechtigungen definieren, welche Aktionen der Auftraggeber mit der Ressource, der die Richtlinie zugewiesen ist. durchführen kann. Wenn sowohl ressourcenbasierte Richtlinien als auch identitätsbasierte Richtlinien für eine Anfrage gelten, AWS überprüft es alle Richtlinien auf mindestens eine. `Allow` Wenn ressourcenbasierte Richtlinien ausgewertet werden, bestimmt der in der Richtlinie angegebene Haupt-ARN, ob implizite Ablehnung in anderen Richtlinientypen auf die endgültige Entscheidung anwendbar sind.
3. IAM-Berechtigungsgrenzen – Berechtigungsgrenzen sind eine erweiterte Funktionalität, die die maximalen Berechtigungen festlegt, die eine identitätsbasierte Richtlinie einer IAM-Entität (Benutzer oder Rolle) erteilen kann. Wenn Sie eine Berechtigungsgrenze für eine Entität festlegen,

kann die Entität nur die Aktionen durchführen, die sowohl von den identitätsbasierten Richtlinien als auch den Berechtigungsgrenzen erlaubt werden. In einigen Fällen kann eine implizite Verweigerung in einer Berechtigungsgrenze die Berechtigungen nicht bechränken, die von einer ressourcenbasierten Richtlinie gewährt werden. Weitere Informationen finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird weiter unten in diesem Thema.

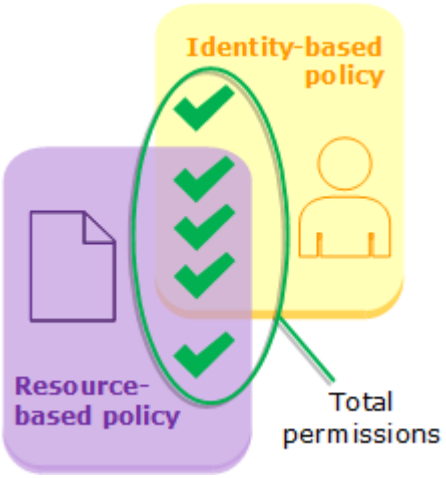
4. AWS Organizations Service Control Policies (SCPs) — SCPs von Organizations geben die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) an. Das SCP-Maximum gilt für Principals in Mitgliedskonten, einschließlich der einzelnen Konten. Root-Benutzer des AWS-Kontos Wenn eine SCP vorhanden ist, erteilen identitätsbasierte und ressourcenbasierte Richtlinien Auftraggeber in Mitgliedskonten nur dann Berechtigungen, wenn diese Richtlinien und die SCP die Aktion zulassen. Wenn sowohl eine Berechtigungsgrenze als auch eine SCP vorhanden sind, muss die Grenze, die SCP und die identitätsbasierte Richtlinie die Aktion zulassen.
5. Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer programmgesteuert erstellen. Um eine Rollensitzung programmgesteuert zu erstellen, verwenden Sie eine der `AssumeRole*`-API-Operationen. Wenn Sie dies tun und Richtlinien für die Sitzung übergeben, sind die resultierenden Sitzungsberechtigungen eine Schnittmenge der identitätsbasierten Richtlinie der IAM-Entität und der Sitzungsrichtlinien. Um eine Sitzung eines Verbundbenutzers zu erstellen, verwenden Sie die Zugriffsschlüssel eines IAM-Benutzers, um den API-Vorgang `GetFederationToken` programmatisch aufzurufen. Eine ressourcenbasierte Richtlinie hat andere Auswirkungen auf die Auswertung der Sitzungsrichtlinienberechtigungen. Der Unterschied hängt davon ab, ob der ARN des Benutzers oder der Sitzung in der ressourcenbasierten Richtlinie als Auftraggeber aufgeführt wird. Weitere Informationen finden Sie unter Sitzungsrichtlinien.

Denken Sie daran, dass eine explizite Zugriffsverweigerung in all diesen Richtlinien eine Zugriffserlaubnis überschreibt.

Auswerten identitätsbasierter Richtlinien mit ressourcenbasierten Richtlinien

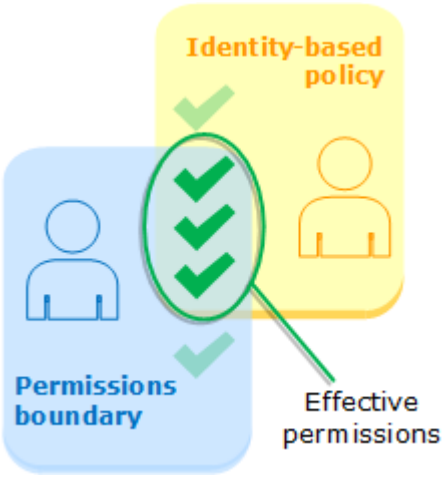
Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien erteilen Identitäten oder Ressourcen, an die sie angefügt sind, Berechtigungen. Wenn eine IAM-Entität (Benutzer oder Rolle) Zugriff auf eine Ressource innerhalb desselben Kontos anfordert, werden alle Berechtigungen AWS bewertet, die durch die identitäts- und ressourcenbasierten Richtlinien gewährt wurden. Die so entstehenden Berechtigungen ergeben sich aus den gesamten Berechtigungen der beiden Typen. Wenn eine

Aktion durch eine identitätsbasierte Richtlinie, eine ressourcenbasierte Richtlinie oder beides zulässig ist, ist die Aktion zulässig. AWS Eine explizite Zugriffsverweigerung in einer der beiden Richtlinien überschreibt die Zugriffserlaubnis.



Auswerten von identitätsbasierten Richtlinien mit Berechtigungsgrenzen

Bei der AWS Auswertung der identitätsbasierten Richtlinien und der Berechtigungsgrenzen für einen Benutzer stellen die resultierenden Berechtigungen die Schnittmenge der beiden Kategorien dar. Das heißt, wenn Sie einem Benutzer mit bestehenden identitätsbasierten Richtlinien eine Berechtigungsgrenze hinzufügen, reduzieren Sie möglicherweise die Aktionen, die der Benutzer ausführen kann. Wenn Sie andererseits eine Berechtigungsgrenze von einem Benutzer entfernen, können Sie die Aktionen erweitern, die dieser ausführen kann. Eine explizite Zugriffsverweigerung in einer der beiden Richtlinien überschreibt die Zugriffserlaubnis. Informationen darüber, wie andere Richtlinientypen mit Berechtigungsgrenzen ausgewertet werden, finden Sie unter Auswerten von effektiven Berechtigungen mit Grenzen.



Bewertung von identitätsbasierten Richtlinien mit Organizations-SCPs

Wenn ein Benutzer zu einem Konto gehört, das Mitglied einer Organisation ist, bilden die resultierenden Berechtigungen eine Schnittmenge aus den Richtlinien des Benutzers und der SCP. Das bedeutet, dass eine Aktion sowohl von der identitätsbasierten Richtlinie als auch von der SCP erlaubt sein muss. Eine explizite Zugriffsverweigerung in einer der beiden Richtlinien überschreibt die Zugriffserlaubnis.

Sie können erfahren ob Ihr Konto als Mitgliedskonto einer Organisation in AWS Organizations eingerichtet ist. Eine SCP hat möglicherweise Auswirkungen auf Mitglieder einer Organisation. Um diese Daten mithilfe des AWS CLI Befehls oder der AWS API-Operation anzuzeigen, müssen Sie über die Berechtigungen für die `organizations:DescribeOrganization` Aktion für Ihre Organisationseinheit verfügen. Sie müssen über zusätzliche Berechtigungen verfügen, um die Operation in der Organizations-Konsole auszuführen. Um zu erfahren, ob ein SCP den Zugriff auf eine bestimmte Anfrage verweigert, oder um Ihre aktuellen Berechtigungen zu ändern, wenden Sie sich an Ihren AWS Organizations Administrator.

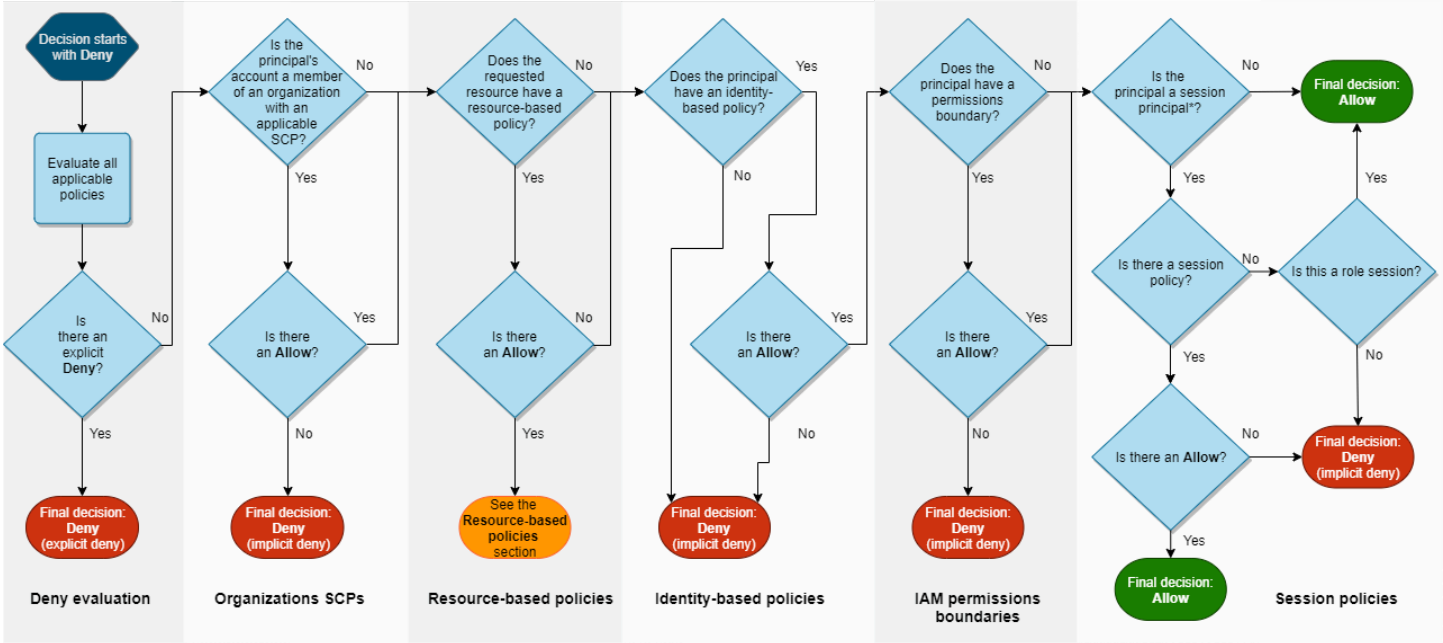
Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird

Gehen Sie davon aus, dass ein Principal eine Anfrage für den AWS Zugriff auf eine Ressource sendet, die sich in demselben Konto wie die Entität des Prinzipals befindet. Der AWS Vollstreckungscode entscheidet, ob der Antrag zugelassen oder abgelehnt werden soll. AWS bewertet alle Richtlinien, die für den Anforderungskontext gelten. Im Folgenden finden Sie eine Zusammenfassung der AWS Bewertungslogik für Richtlinien innerhalb eines einzelnen Kontos.

- Standardmäßig werden alle Anfragen implizit abgelehnt, mit Ausnahme von Root-Benutzer des AWS-Kontos, die vollen Zugriff hat.

- Eine explizite Zugriffserlaubnis in einer identitätsbasierten oder ressourcenbasierten Richtlinie hat Vorrang vor diesem Standardwert.
- Wenn eine Berechtigungsgrenze, Organizations-SCP oder Sitzungsrichtlinie vorhanden ist, kann sie die Zugriffserlaubnis mit einer impliziten Zugriffsverweigerung überschreiben.
- Eine explizite Zugriffsverweigerung überschreibt jede Zugriffserlaubnis in einer Richtlinie.

Das folgende Flussdiagramm enthält einen Überblick über den Entscheidungsprozess. Dieses Flussdiagramm deckt nicht die Auswirkungen ressourcenbasierter Richtlinien und impliziter Ablehnungen in anderen Arten von Richtlinien ab.



*A session principal is either a role session or an IAM federated user session.

1. Auswertung für eine Zugriffsverweigerung – Standardmäßig werden alle Anforderungen verweigert. Dieser Vorgang wird als implizite Zugriffsverweigerung bezeichnet. Der AWS Durchsetzungscode bewertet alle Richtlinien innerhalb des Kontos, die für die Anfrage gelten. Dazu gehören AWS Organizations SCPs, ressourcenbasierte Richtlinien, identitätsbasierte Richtlinien, IAM-Berechtigungsgrenzen und Sitzungsrichtlinien. Der Durchführungscode sucht in allen diesen Richtlinien nach einer `Deny`-Anweisung, die für die Anforderung gilt. Dieser Vorgang wird als explizite Zugriffsverweigerung bezeichnet. Wenn der Code auch nur eine gültige explizite Zugriffsverweigerung findet, wird final Zugriffsverweigerung zurückgegeben und die Anforderung wird abgelehnt. Wenn es keine ausdrückliche Verweigerung gibt, wird die Auswertung des Erzwingungscodes fortgesetzt.
2. SCPs für Organizations — Anschließend bewertet der Durchsetzungscode die AWS Organizations Service Control Policies (SCPs), die für die Anfrage gelten. SCPs gelten für Auftraggeber des

Kontos, dem die SCPs zugewiesen sind. Wenn der Durchführungscode keine gültigen `Allow`-Anweisungen in den SCPs findet, wird die Anforderung explizit abgelehnt, selbst wenn die Ablehnung implizit ist. Der Code gibt die finale Entscheidung Zugriffsverweigerung zurück. Wenn keine SCP vorhanden ist oder wenn die SCP die angeforderte Aktion zulässt, wird die Erzwingungscodeauswertung fortgesetzt.

3. Ressourcenbasierte Richtlinien - Innerhalb desselben Kontos wirken sich ressourcenbasierte Richtlinien unterschiedlich aus, abhängig von der Art des Prinzipals, der auf die Ressource zugreift, und dem Prinzipal, der in der ressourcenbasierten Richtlinie zulässig ist. Abhängig von der Art des Prinzipals, kann ein `Allow` in einer ressourcenbasierten Richtlinie zu einer endgültigen Entscheidung von `Allow` führen, auch wenn eine implizite Ablehnung in einer identitätsbasierten Richtlinie, Berechtigungsgrenze oder Sitzungsrichtlinie vorhanden ist.

   Für die meisten Ressourcen benötigen Sie nur eine explizite Genehmigung für den Prinzipal entweder in einer identitätsbasierten Richtlinie oder in einer ressourcenbasierten Richtlinie, um Zugriff zu gewähren. IAM-Rollenvertrauensrichtlinien und KMS-Schlüsselrichtlinien sind Ausnahmen von dieser Logik, da sie den Zugriff für Auftraggeber ausdrücklich zulassen müssen.

   Die ressourcenbasierte Richtlinienlogik unterscheidet sich von anderen Richtlinientypen, wenn der angegebene Prinzipal ein IAM-Benutzer, eine IAM-Rolle oder ein Sitzungsprinzipal ist. Zu den Sitzungsprinzipalen gehören IAM-Rollensitzungen oder ein IAM-Verbundbenutzer-Sitzung. Wenn eine ressourcenbasierte Richtlinie dem IAM-Benutzer oder dem Sitzungssprinzipal, der die Anforderung stellt, die Berechtigung direkt erteilt, wirkt sich eine implizite Ablehnung in einer identitätsbasierten Richtlinie, einer Berechtigungsgrenze oder einer Sitzungsrichtlinie nicht auf die endgültige Entscheidung aus.

   Die folgende Tabelle hilft Ihnen, die Auswirkungen ressourcenbasierter Richtlinien für verschiedene Haupttypen zu verstehen, wenn implizite Ablehnungen in identitätsbasierten Richtlinien, Berechtigungsgrenzen und Sitzungsrichtlinien implizite Ablehnungen vorhanden sind.

Ressourcenbasierte Richtlinien und implizite Verweigerungen in anderen Richtlinientypen (dasselbe Konto)

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
IAM-Rolle	Nicht zutreffend	Nicht zutreffend	Nicht zutreffend	Nicht zutreffend	Nicht zutreffend	Eine Rolle selbst kann keine Anfrage stellen. Anfragen werden mit der Rollensitzung gestellt, nachdem eine Rolle angenommen wurde.

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
IAM-Rollensitzung	Erlaubt - Rollen-ARN	Implizite Verweigerung	Implizite Verweigerung	Implizite Verweigerung	DENY	Berechtigungsgrenze und Sitzungsrichtlinie werden im Rahmen der endgültigen Entscheidung ausgewertet. Eine implizite Ablehnung in beiden Richtlinien führt zu einer VERWEIGERN-Entscheidung.

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
IAM-Rollensitzung	Erlaubt Rollensitzungs-ARN	Implizite Verweigerung	Implizite Verweigerung	Implizite Verweigerung	ERLAUBEN	Berechtigungen werden direkt für die Sitzung erteilt. Andere Richtlinientypen haben keinen Einfluss auf die Entscheidung.

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
IAM-Benutzer	Ermöglicht ARN für IAM-Benutzer	Implizite Verweigerung	Implizite Verweigerung	Nicht zutreffend	ERLAUBEN	Berechtigungen werden direkt an den Benutzer erteilt. Andere Richtlinientypen haben keinen Einfluss auf die Entscheidung.
IAM-Verbundbenutzer (`GetFederationToken`)	Ermöglicht ARN für IAM-Benutzer	Implizite Verweigerung	Implizite Verweigerung	Implizite Verweigerung	DENY	Eine implizite Verweigerung in der Berechtigungsgrenze oder in der Sitzungsrichtlinie führt zu einem VERWEIGERN.

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
IAM-Verbundbenutzer (`GetFederationToken`)	Ermöglicht ARN für IAM-Verbundbenutzersitzungen	Implizite Verweigerung	Implizite Verweigerung	Implizite Verweigerung	ERLAUBEN	Berechtigungen werden direkt für die Sitzung erteilt. Andere Richtlinientypen haben keinen Einfluss auf die Entscheidung.

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
Root-Benutzer	Ermöglicht ARN für Stammbenutzer	Nicht zutreffend	Nicht zutreffend	Nicht zutreffend	ERLAUBEN	Das Root-Konto hat vollständigen und uneingeschränkten Zugriff auf alle Ressourcen in Ihrem AWS-Konto. Um zu erfahren, wie Sie den Zugriff auf Root-Benutzer für Konten in AWS Organizations kontrollieren können, finden Sie unter Service-Kontrollrichtlinien (SCPs) im

Prinzipal, der die Anforderung stellt	Ressourcenbasierte Richtlinie	Identitätsbasierte Richtlinien	Berechtigungsgrenze	Sitzungsrichtlinie	Ergebnis	Grund
						Benutzerhandbuch für Organisationen.
AWS Dienstprinzipal	Erlaubt einen AWS Dienstprinzipal	Nicht zutreffend	Nicht zutreffend	Nicht zutreffend	ERLAUBEN	Wenn eine ressourcenbasierte Richtlinie Berechtigungen direkt an AWS-Service-Prinzipal erteilen, haben andere Richtlinientypen keinen Einfluss auf die Entscheidung.

- IAM-Rolle - Ressourcenbasierte Richtlinien, die Berechtigungen für eine IAM-Rolle ARN erteilen, sind durch eine implizite Verweigerung in einer Berechtigungsgrenze oder einer Sitzungsrichtlinie begrenzt.

  Beispielrolle ARN

```
arn:aws:iam::111122223333:role/examplerole
```

- IAM-Rollensitzung – Innerhalb desselben Kontos gewähren ressourcenbasierte Richtlinien, die einem IAM-Rollensitzungs-ARN Berechtigungen erteilen, auch direkt Berechtigungen für die angenommene Rollensitzung. Berechtigungen, die direkt für eine Sitzung gewährt werden, sind nicht durch eine implizite Verweigerung in einer identitätsbasierten Richtlinie, einer Berechtigungsgrenze oder einer Sitzungsrichtlinie beschränkt. Wenn Sie eine Rolle übernehmen und eine Anfrage stellen, ist der Prinzipal, der die Anfrage stellt, der ARN der IAM-Rollensitzung und nicht der ARN der Rolle selbst.

  Beispiel für eine Rollensitzung ARN

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- IAM-Benutzer - Innerhalb desselben Kontos sind ressourcenbasierte Richtlinien, die einem IAM-Benutzer-ARN (das kein Verbundbenutzersitzung ist) Berechtigungen erteilt, nicht durch eine implizite Verweigerung in einer identitätsbasierten Richtlinie oder Berechtigungsgrenze beschränkt.

  Beispiel eines IAM-Benutzer-ARN

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- IAM-Verbundbenutzersitzung – Eine IAM-Verbundbenutzersitzung ist eine Sitzung, die durch Aufruf von `GetFederationToken` erstellt wurde. Wenn ein Verbundbenutzer eine Anfrage stellt, ist der Prinzipal, der die Anfrage stellt, der ARN des Verbundbenutzers und nicht der ARN des IAM-Benutzers, der den Verbund erstellt hat. Innerhalb desselben Kontos erteilen ressourcenbasierte Richtlinien, die einem Verbundbenutzer-ARN Berechtigungen gewähren, der Sitzung direkt Berechtigungen. Berechtigungen, die direkt für eine Sitzung gewährt werden, sind nicht durch eine implizite Verweigerung in einer identitätsbasierten Richtlinie, einer Berechtigungsgrenze oder einer Sitzungsrichtlinie beschränkt.

  Wenn jedoch eine ressourcenbasierte Richtlinie dem ARN des IAM-Benutzers, der den Verbund erstellt hat, die Berechtigung erteilt, werden Anfragen des Verbundbenutzers während der Sitzung durch eine implizite Verweigerung in einer Berechtigungsgrenze oder Sitzungsrichtlinie eingeschränkt.

  Beispiel für IAM-Verbundbenutzersitzung ARN

```
arn:aws:sts::111122223333:federated-user/exampleuser
```

4. Identitätsbasierte Richtlinien – Der Code überprüft dann die identitätsbasierten Richtlinien auf den Auftraggeber. Für einen IAM-Benutzer gehören hierzu Benutzerrichtlinien und Richtlinien von Gruppen, zu denen der Benutzer gehört. Wenn es keine Anweisungen gibt, die die angeforderte Aktion zulassen, dann wird die Anforderung implizit verweigert und der Code gibt die finale Entscheidung Deny (Zugriffsverweigerung) zurück. Wenn eine Anweisung in einer geltenden identitätsbasierten Richtlinie die angeforderte Aktion zulässt, dann wird der Code weitergeführt.
5. IAM-Berechtigungsgrenzen - Der Durchführungscode überprüft dann, ob die IAM-Entität, die vom Auftraggeber verwendet wird, eine Berechtigungsgrenze aufweist. Wenn die Richtlinie, die verwendet wird, um die Berechtigungsgrenze festzulegen, die angeforderte Aktion nicht zulässt, dann wird die Anforderung implizit verweigert. Der Code gibt die finale Entscheidung Deny (Zugriffsverweigerung) zurück. Wenn es keine Berechtigungsgrenze gibt oder wenn die Berechtigungsgrenze die angeforderte Aktion zulässt, wird der Code fortgesetzt.
6. Sitzungsrichtlinien – Der Code überprüft dann, ob der Prinzipal ein Sitzungsprinzipal ist. Sitzungsprinzipale umfassen eine IAM-Rollensitzung oder eine IAM-Verbundbenutzersitzung. Wenn der Prinzipal kein Sitzungsprinzipal ist, gibt der Durchsetzungscode eine endgültige Entscheidung vonErlauben zurück.

   Für Sitzungsprinzipale, der Code prüft, ob eine Sitzungsrichtlinie in der Anforderung übergeben wurde. Sie können eine Sitzungsrichtlinie übergeben, während Sie die AWS API AWS CLI oder verwenden, um temporäre Anmeldeinformationen für eine Rolle oder einen IAM-Verbundbenutzer abzurufen.

   - Wenn die Sitzungsrichtlinie vorhanden ist und die angeforderte Aktion nicht zulässt, dann wird die Anforderung implizit verweigert. Der Code gibt die finale Entscheidung Deny (Zugriffsverweigerung) zurück.
   - Wenn keine Sitzungsrichtlinie vorhanden ist, prüft der Code, ob der Prinzipal eine Rollensitzung ist. Wenn der Prinzipal eine Rollensitzung ist, lautet die Anforderung Erlaubt. Daher wird die Anforderung implizit abgelehnt und der Code gibt eine endgültige Entscheidung von Verweigern zurück.
   - Wenn eine Sitzungsrichtlinie vorhanden ist und die angeforderte Aktion erlaubt, dann gibt der Durchführungscode eine endgültige Entscheidung von Erlauben zurück.
7. Fehler — Wenn der AWS Erzwingungscode zu irgendeinem Zeitpunkt während der Evaluierung auf einen Fehler stößt, generiert er eine Ausnahme und wird geschlossen.

Beispielauswertung identitätsbasierter Richtlinien und ressourcenbasierter Richtlinien

Die gebräuchlichsten Richtlinientypen sind identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien. Wenn Zugriff auf eine Ressource angefordert wird, werden alle Berechtigungen AWS ausgewertet, die durch die Richtlinien für mindestens eine Option „Zulassen“ innerhalb desselben Kontos gewährt wurden. Eine explizite Zugriffsverweigerung in einer der Richtlinien setzt eine Zugriffserlaubnis außer Kraft.

> **Important**
>
> Wenn entweder die identitätsbasierte Richtlinie oder die ressourcenbasierte Richtlinie innerhalb desselben Kontos die Anforderung zulässt und die andere nicht, ist die Anforderung trotzdem zulässig.

Angenommen, Carlos hat den Benutzernamen `carlossalazar`, und er versucht, eine Datei im Amazon S3-Bucket `carlossalazar-logs` zu speichern.

Außerdem wird davon ausgegangen, dass die folgende Richtlinie dem IAM-Benutzer `carlossalazar` zugeordnet ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
```

```
                "arn:aws:s3:::carlossalazar"
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

Die `AllowS3ListRead` -Anweisung in dieser Richtlinie erlaubt Carlos, eine Liste aller Buckets im Konto anzuzeigen. Die `AllowS3Self`-Anweisung erlaubt Carlos vollständigen Zugriff auf den Bucket mit demselben Namen wie seinem Benutzernamen. Die `DenyS3Logs`-Anweisung verweigert Carlos den Zugriff auf S3-Buckets mit der Zeichenfolge `log` im Namen.

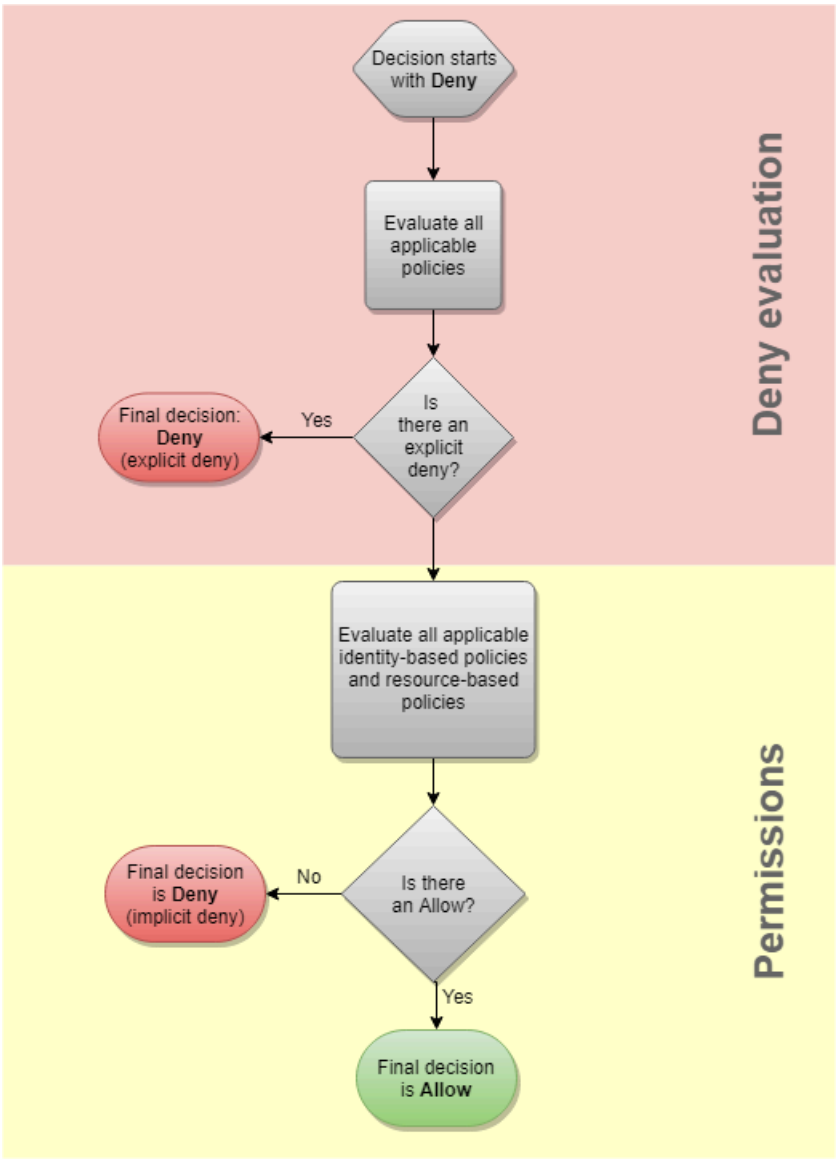
Außerdem ist die folgende ressourcenbasierte Richtlinie (eine sogenannte Bucket-Richtlinie) dem Bucket `carlossalazar` zugeordnet.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
        }
    ]
}
```

Diese Richtlinie gibt an, dass nur der Benutzer `carlossalazar` auf den Bucket `carlossalazar` zugreifen kann.

Wenn Carlos seine Anfrage zum Speichern einer Datei im `carlossalazar-logs` Bucket AWS stellt, bestimmt er, welche Richtlinien für die Anfrage gelten. In diesem Fall gelten

nur die identitätsbasierte Richtlinie und die ressourcenbasierte Richtlinie. Beides sind Berechtigungsrichtlinien. Da keine Berechtigungsgrenzen gelten, wird die Auswertungslogik auf die folgende Logik reduziert.



AWS sucht zunächst nach einer `Deny` Aussage, die für den Kontext der Anfrage gilt. Es findet einen, weil die identitätsbasierte Richtlinie Carlos explizit den Zugriff auf alle S3-Buckets verweigert, die für die Protokollierung verwendet werden. Carlos wird der Zugriff verweigert.

Gehen Sie davon aus, dass er dann seinen Fehler erkennt und versucht, die Datei im `carlossalazar` Bucket zu speichern. AWS sucht nach einer `Deny` Aussage und findet keine. Dann überprüft es die Berechtigungsrichtlinien. Sowohl die identitätsbasierte Richtlinie, als auch die ressourcenbasierte Richtlinie lassen die Anforderung zu. AWS Erlaubt daher die Anfrage. Hätte eine von ihnen die Anweisung ausdrücklich abgelehnt, wäre die Anforderung abgelehnt worden. Wenn

einer der Richtlinientypen die Anforderung zulässt und der andere nicht, ist die Anforderung weiterhin zulässig.

Der Unterschied zwischen expliziten und impliziten Zugriffsverweigerungen

Eine Anforderung führt zu einer expliziten Zugriffsverweigerung, wenn eine anwendbare Richtlinie eine `Deny`-Anweisung enthält. Wenn Richtlinien, die auf eine Anforderung anwendbar sind, eine `Allow`-Anweisung und eine `Deny`-Anweisung enthalten, übersteuert die `Deny` -Anweisung die `Allow` -Anweisung. Die Anforderung wird abgelehnt.

Eine implizite Verweigerung tritt auf, wenn es keine entsprechende `Deny`-Anweisung, aber auch keine anwendbare `Allow`-Anweisung gibt. Da einem IAM-Prinzipal standardmäßig der Zugriff verweigert wird, muss ihm explizit erlaubt werden, eine Aktion auszuführen. Andernfalls wird ihnen der Zugriff implizit verweigert.

Bei der Planung Ihrer Autorisierungsstrategie müssen Sie Richtlinien mit `Allow`-Anweisungen erstellen, um Ihren Auftraggeber zu gestatten, erfolgreich Anforderungen durchzuführen. Sie können jedoch eine beliebige Kombination von expliziten und impliziten Zugriffsverweigerungen wählen.

Sie können beispielsweise die folgende Richtlinie erstellen, die zulässige Aktionen, implizit verweigerte Aktionen und explizit verweigerte Aktionen enthält. Die `AllowGetList`-Aussage erlaubt Lesezugriff auf IAM-Aktionen, die mit den Präfixen `Get` und `List` beginnen. Alle anderen Aktionen in IAM, wie `iam:CreatePolicy` sind implizit abgelehnt. Die `DenyReports`-Aussage verweigert explizit Zugriff auf IAM-Berichte durch Verweigern des Zugriffs auf Aktionen, die den `Report`-Suffix, wie `iam:GetOrganizationsAccessReport` enthalten. Wenn jemand diesem Prinzipal eine andere Richtlinie hinzufügt, um ihm Zugriff auf IAM-Berichte zu gewähren, z. B. `iam:GenerateCredentialReport`, werden berichtsbezogene Anfragen aufgrund dieser ausdrücklichen Ablehnung immer noch abgelehnt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Logik für die kontenübergreifende Richtlinienauswertung

Sie können einem Auftraggeber in einem Konto gestatten, auf Ressourcen in einem zweiten Konto zuzugreifen. Dies wird als kontenübergreifender Zugriff bezeichnet. Wenn Sie kontenübergreifenden Zugriff zulassen, wird das Konto, in dem der Auftraggeber existiert, als vertrauenswürdiges-Konto bezeichnet. Das Konto, in dem die Ressource existiert, ist das vertrauende Konto.

Um den kontenübergreifenden Zugriff zu ermöglichen, weisen Sie eine ressourcenbasierte Richtlinie für die freizugebende Ressource zu. Sie müssen außerdem eine identitätsbasierte Richtlinie für die Identität zuweisen, die in der Anforderung als Prinzipal fungiert. Die ressourcenbasierte Richtlinie im vertrauenden Konto muss den Auftraggeber des vertrauenswürdigen Kontos angeben, der Zugriff auf die Ressource hat. Sie können das gesamte Konto oder seine IAM-Benutzer, Verbundbenutzer, -Rollen oder Sitzungen mit angenommener Rolle angeben. Sie können auch einen AWS Dienst als Principal angeben. Weitere Informationen finden Sie unter Angeben eines Auftraggebers.

Die identitätsbasierte Richtlinie des Auftraggebers muss den angeforderten Zugriff auf die Ressource im vertrauenden Service zulassen. Sie können dies tun, indem Sie den ARN der Ressource angeben oder indem Sie den Zugriff auf alle Ressourcen zulassen (*).

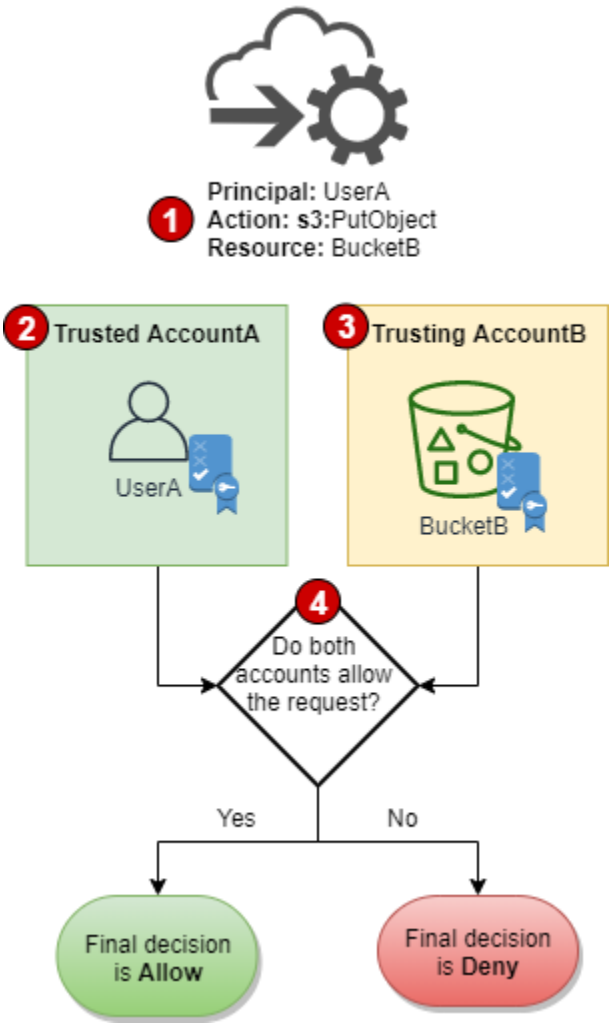
In können Sie eine ressourcenbasierte Richtlinie einer IAM-Rolle zuordnen, um Auftraggeber in anderen Konten diese Rolle übernehmen zu lassen. Die ressourcenbasierte Richtlinie der Rolle wird als Rollenvertrauensrichtlinie bezeichnet. Nach dem Übernehmen dieser Rolle können die zugelassenen Auftraggeber die resultierenden temporären Anmeldeinformationen verwenden, um auf mehrere Ressourcen in Ihrem Konto zuzugreifen. Dieser Zugriff wird in der identitätsbasierten Berechtigungsrichtlinie der Rolle definiert. Informationen darüber, wie sich das Zulassen von kontenübergreifendem Zugriff mit Rollen von dem Zulassen von kontenübergreifendem Zugriff mit anderen ressourcenbasierten Richtlinien unterscheidet, finden Sie unter Kontoübergreifender Zugriff auf Ressourcen in IAM.

> **Important**
>
> Andere Services können sich auf die Richtlinienauswertungslogik auswirken. AWS Organizations Unterstützt beispielsweise Richtlinien zur Dienststeuerung, die auf ein oder mehrere Konten von Prinzipalen angewendet werden können. AWS Resource Access Manager unterstützt Richtlinienfragmente, die steuern, welche Aktionen Principals auf Ressourcen ausführen dürfen, die mit ihnen gemeinsam genutzt werden.

Festlegen, ob eine kontenübergreifende Anforderung zulässig ist

Bei kontenübergreifenden Anforderungen muss der Anforderer in der vertrauenswürdigen `AccountA` eine identitätsbasierte Richtlinie haben. Diese Richtlinie muss ihnen gestatten, eine Anforderung an die Ressource in der vertrauenden `AccountB` zu stellen. Zusätzlich muss die ressourcenbasierte Richtlinie in `AccountB` dem Anforderer in `AccountA` den Zugriff auf die Ressource gestatten.

Wenn Sie eine kontenübergreifende Anfrage stellen, AWS führt zwei Bewertungen durch. AWS bewertet die Anfrage im vertrauenswürdigen Konto und im vertrauenswürdigen Konto. Weitere Informationen darüber, wie eine Anforderung innerhalb eines einzelnen Kontos ausgewertet wird, finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird. Die Anforderung ist nur dann zulässig, wenn beide Auswertungen eine Entscheidung von `Allow` zurückgeben.



1. Wenn ein Auftraggeber in einem Konto eine Anforderung für den Zugriff auf eine Ressource in einem anderen Konto stellt, ist dies eine kontenübergreifende Anforderung.
2. Der anfordernde Auftraggeber ist im vertrauenswürdigen Konto (`AccountA`) vorhanden. Wenn AWS dieses Konto auswertet, prüft es die identitätsbasierte Richtlinie und alle Richtlinien, die eine identitätsbasierte Richtlinie einschränken können. Weitere Informationen finden Sie unter Auswerten von Richtlinien in einem einzelnen Konto.
3. Die angeforderte Ressource ist im vertrauenden Konto (`AccountB`) vorhanden. Wenn AWS dieses Konto auswertet, prüft es die ressourcenbasierte Richtlinie, die der angeforderten Ressource zugeordnet ist, sowie alle Richtlinien, die eine ressourcenbasierte Richtlinie einschränken können. Weitere Informationen finden Sie unter Auswerten von Richtlinien in einem einzelnen Konto.
4. AWS lässt die Anfrage nur zu, wenn beide Bewertungen der Kontorichtlinien die Anfrage zulassen.

Beispiel für kontenübergreifende Richtlinienauswertung

Das folgende Beispiel veranschaulicht ein Szenario, in dem einem Benutzer in einem Konto Berechtigungen durch eine ressourcenbasierte Richtlinie in einem zweiten Konto gewährt werden.

Nehmen wir an, dass Carlos ein Entwickler mit einem IAM-Benutzer namens `carlossalazar` im Konto 111111111111 ist. Er möchte eine Datei im `Production-logs` Amazon S3-Bucket im Konto 222222222222 speichern.

Außerdem wird davon ausgegangen, dass die folgende Richtlinie dem IAM-Benutzer `carlossalazar` zugeordnet ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::Production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

Die `AllowS3ListRead`-Anweisung in dieser Richtlinie erlaubt es Carlos, eine Liste aller Buckets in Amazon S3 anzuzeigen. Die `AllowS3ProductionObjectActions`-Anweisung gewährt Carlos Vollzugriff auf Objekte im `Production`-Bucket. Die `DenyS3Logs`-Anweisung verweigert Carlos

den Zugriff auf S3-Buckets mit der Zeichenfolge `log` im Namen. Außerdem wird der Zugriff auf alle Objekte in diesen Buckets verweigert.

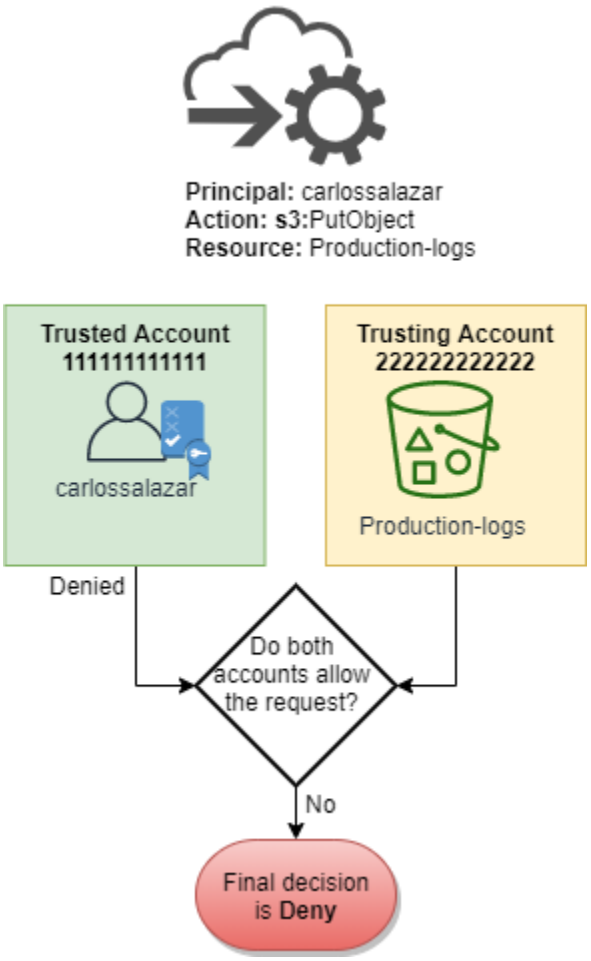
Zusätzlich wird die folgende ressourcenbasierte Richtlinie (eine so genannte Bucket-Richtlinie) an den `Production`-Bucket im Konto 222222222222 angehängt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
            "Resource": "arn:aws:s3:::Production/*"
        }
    ]
}
```

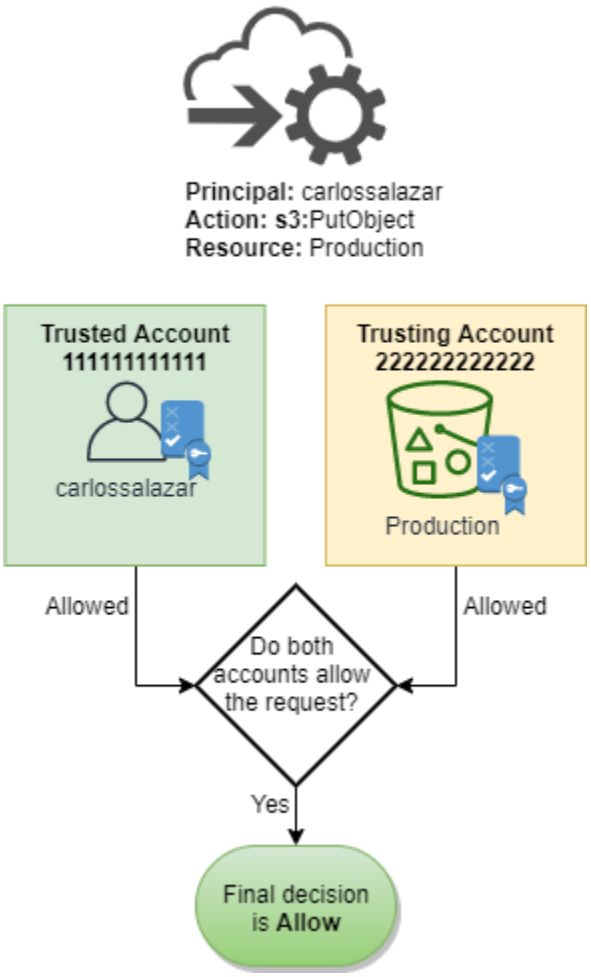
Mit dieser Richtlinie kann der `carlossalazar`-Benutzer auf Objekte im `Production`-Bucket zugreifen. Er kann die Objekte im Bucket erstellen und bearbeiten, aber nicht löschen. Er kann den Bucket nicht selbst verwalten.

Wenn Carlos seine Anforderung sendet, eine Datei im Bucket `Production-logs` zu speichern, bestimmt AWS , welche Richtlinien für die Anforderung gelten. In diesem Fall ist die identitätsbasierte Richtlinie, die Benutzer `carlossalazar` zugeordnet ist, die einzige Richtlinie, die in Konto `111111111111` gilt. In Konto `222222222222` ist keine ressourcenbasierte Richtlinie für den `Production-logs`-Bucket festgelegt. Wenn AWS das Konto `111111111111` auswertet, gibt es die Entscheidung `Deny` zurück. Das liegt daran, dass die `DenyS3Logs`-Anweisung in der identitätsbasierten Richtlinie den Zugriff auf alle Log-Buckets explizit verweigert. Weitere Informationen darüber, wie eine Anforderung innerhalb eines einzelnen Kontos ausgewertet wird, finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird.

Da die Anforderung innerhalb eines der Konten explizit verweigert wird, besteht die endgültige Entscheidung darin, die Anforderung zu verweigern.



Gehen Sie davon aus, dass Carlos dann seinen Fehler erkennt und versucht, die Datei im `Production` Bucket zu speichern. AWS überprüft zunächst das Konto`111111111111`, um festzustellen, ob die Anfrage zulässig ist. Es gilt nur die identitätsbasierte Richtlinie, die die Anfrage zulässt. AWS überprüft dann das Konto. `222222222222` Nur die ressourcenbasierte Richtlinie, die dem `Production`-Bucket zugeordnet ist, gilt und lässt die Anforderung zu. Da beide Konten die Anforderung zulassen, besteht die endgültige Entscheidung darin, die Anforderung zuzulassen.



Grammatik der IAM-JSON-Richtliniensprache

Auf dieser Seite wird eine formelle Grammatik der Sprache vorgestellt, die zum Erstellen von JSON-Richtlinien in IAM verwendet wird. Sie müssen diese Grammatik beherrschen, um Richtlinien erstellen und validieren zu können.

Beispielrichtlinien finden Sie in den folgenden Themen:

- Berechtigungen und Richtlinien in IAM
- Beispiele für identitätsbasierte Richtlinien in IAM
- Beispielrichtlinien für die Arbeit in der Amazon EC2 EC2-Konsole und Beispielrichtlinien für die Arbeit mit der AWS CLI, der Amazon EC2 EC2-CLI oder einem AWS SDK im Amazon EC2 EC2-Benutzerhandbuch.
- Beispiele für Bucket-Richtlinien und Nutzer-Richtlinien finden Sie im Benutzerhandbuch für Amazon Simple Storage Service.

Beispiele für Richtlinien, die in anderen AWS Diensten verwendet werden, finden Sie in der Dokumentation zu diesen Diensten.

Themen

Die Richtliniensprache und JSON

Richtlinien werden in JSON ausgedrückt. Wenn Sie eine JSON-Richtlinie erstellen oder bearbeiten, kann IAM eine Richtlinienvalidierung durchführen, um Ihnen beim Erstellen einer effektiven Richtlinie zu helfen. IAM identifiziert JSON-Syntaxfehler, während IAM Access Analyzer zusätzliche Richtlinienüberprüfungen mit Empfehlungen zur weiteren Verfeinerung Ihrer Richtlinien bietet. Weitere Informationen zur Richtlinienvalidierung finden Sie unter Validieren von IAM-Richtlinien. Weitere Informationen zu IAM Access Analyzer Richtlinienvalidierungen und umsetzbaren Empfehlungen finden Sie unter IAM Access Analyzer-Richtlinienvalidierung.

Dieses Dokument enthält keine umfassende Beschreibung gültiger JSON. Nachfolgend finden Sie jedoch einige grundlegende JSON-Regeln:

- Leerzeichen zwischen einzelnen Einheiten sind zulässig.
- Werte müssen in Anführungszeichen stehen. Anführungszeichen sind für nummerische und boolesche Werte optional.
- Viele Elemente wie z. B. `action_string_list` und `resource_string_list` können ein JSON-Array als Wert enthalten. Arrays können einen oder mehrere Werte enthalten. Wenn ein Array mehrere Werte enthält, steht es in eckigen Klammern (`[` und `]`), die einzelnen Werte sind durch Komma voneinander getrennt. Beispiel:

  ```
  "Action" : ["ec2:Describe*","ec2:List*"]
  ```

- Die grundlegenden JSON-Datentypen (Boolescher Wert, Zahl und Zeichenfolge) sind in RFC 7159 definiert.

In dieser Grammatik verwendete Konventionen

In dieser Grammatik werden folgende Konventionen verwendet:

- Die folgenden Zeichen sind JSON-Token und sind in Richtlinien enthalten:

  ```
  { } [ ] " , :
  ```
- Die folgenden Zeichen sind Sonderzeichen in der Grammatik und sind nicht in Richtlinien enthalten:

  ```
  = < > ( ) |
  ```
- Wenn für ein Element mehrere Werte zulässig sind, wird dies durch wiederholte Werte, ein Komma als Trennzeichen und eine Ellipse (`...`) dargestellt. Beispiele:

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Wenn mehrere Werte zulässig sind, ist es auch zulässig, nur einen Wert anzugeben. Wenn nur ein Wert angegeben wird, muss das anschließende Komma weggelassen werden. Wenn ein Element ein Array (durch [und] gekennzeichnet) enthalten kann, das jedoch nur einen Wert enthält, können die Klammern weggelassen werden. Beispiele:

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```
- Ein Fragezeichen (?) nach einem Element gibt an, dass das Element optional ist. Beispiel:

  ```
  <version_block?>
  ```

  Weitere Informationen zu optionalen Elementen können Sie den Hinweisen nach jedem Grammatikelement entnehmen.
- Eine vertikale Linie (`|`) zwischen Elemente gibt Alternativen an. Runde Klammern geben den Umfang der Alternativen an. Beispiel:

  ```
  ("Principal" | "NotPrincipal")
  ```
- Elemente, bei denen es sich um Zeichenfolgen handelt, müssen von doppelten Anführungszeichen (") umschlossen sein. Beispiel:

  ```
  <version_block> = "Version" : ("2008-10-17" | "2012-10-17")
  ```

Weitere Hinweise finden Sie unter Hinweise zur Richtliniengrammatik in der Grammatikauflistung.

Grammatik

Die folgende Liste beschreibt die Grammatik der Richtliniensprache. Die in der Liste verwendeten Konventionen wurden im vorherigen Abschnitt beschrieben. Zusätzliche Informationen finden Sie in den nachfolgenden Hinweisen.

> **Note**
>
> Diese Grammatik beschreibt Richtlinien der Versionen `2008-10-17` und `2012-10-17`. Das Richtlinienelement `Version` unterscheidet sich von einer Richtlinienversion. Das Richtlinienelement `Version` wird innerhalb einer Richtlinie verwendet und gibt die Version der Richtliniensprache an. Andererseits wird eine Richtlinienversion erstellt, wenn Sie in IAM eine benutzerdefinierte, verwaltete Richtlinie bearbeiten. Die vorhandene Richtlinie wird von der geänderten Richtlinie nicht überschrieben. IAM erstellt stattdessen eine neue Version der verwalteten Richtlinie. Weitere Informationen zum Richtlinienelement `Version` finden Sie unter IAM-JSON-Richtlinienelemente: Version. Weitere Informationen zu den Richtlinienversionen finden Sie unter the section called “Versioning von IAM-Richtlinien”.

```
policy  = {
     <version_block?>
     <id_block?>
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17" | "2012-10-17")

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
```

```
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Hinweise zur Richtliniengrammatik

- Eine einzelne Richtlinie kann eine Reihe von Anweisungen enthalten.
- Die maximale Größe für Richtlinien beträgt je nach der Entität, der die Richtlinie zugeordnet ist, zwischen 2.048 und 10.240 Zeichen. Weitere Informationen finden Sie unter IAM und Kontingente AWS STS. Bei der Berechnung der Richtliniengröße werden Leerzeichen nicht mitgezählt.
- Einzelne Elemente dürfen nicht mehrere Instances desselben Schlüssels enthalten. Sie können beispielsweise den Block `Effect` nicht zweimal innerhalb derselben Anweisung verwenden.
- Blöcke können in beliebiger Reihenfolge angegeben werden. `version_block` kann beispielsweise in einer Richtlinie von `id_block` gefolgt werden. Ebenso können die Blöcke `effect_block`, `principal_block` und `action_block` in einer Anweisung in beliebiger Reihenfolge angegeben werden.

- Der Block `id_block` ist in ressourcenbasierten Richtlinien optional. Er darf nicht in identitätsbasierten Richtlinien enthalten sein.
- Das Element `principal_block` muss in ressourcenbasierten Richtlinien (z. B. in Amazon S3-Bucket-Richtlinien) sowie in Vertrauensrichtlinien für IAM-Rollen enthalten sein. Er darf nicht in identitätsbasierten Richtlinien enthalten sein.
- Das `principal_map`-Element in Amazon S3-Bucket-Richtlinien kann die `CanonicalUser`-ID enthalten. Der Großteil der ressourcenbasierten Richtlinien unterstützt dieses Mapping nicht. Weitere Informationen zur Verwendung der kanonischen Benutzer-ID in einer Bucket-Richtlinie finden Sie unter Angeben eines Auftraggebers in einer Richtlinie im Benutzerhandbuch für Amazon Simple Storage Service.
- Für Zeichenfolgewerte (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string` sowie die Zeichenfolgenversion in `condition_value`) kann es jeweils eigene Mindest- und Höchstlängen, bestimmte zulässige Werte oder ein erforderliches internes Format geben.

Hinweise zu Zeichenfolgewerten

In diesem Abschnitt werden Zeichenfolgewerte, die in verschiedenen Elementen von Richtlinien verwendet werden, detailliert beschrieben.

`action_string`

Besteht aus einem Service-Namespace, einem Doppelpunkt und dem Namen einer Aktion. Aktionsnamen können Platzhalterzeichen enthalten. Beispiele:

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"

"Action":[
  "s3:Get*",
  "s3:List*"
```

```
]
```

policy_id_string

Bietet eine Möglichkeit, Informationen zur Richtlinie im Ganzen bereitzustellen. Einige Services wie Amazon SQS und Amazon SNS verwenden das reservierte Element `Id`. Sofern policy_id_string nicht von einzelnen Services anderweitig beschränkt ist, sind Leerzeichen zulässig. Für einige Services muss dieser Wert innerhalb eines AWS -Kontos eindeutig sein.

> **Note**
>
> Der Block `id_block` ist in ressourcenbasierten Richtlinien, nicht jedoch in identitätsbasierten Richtlinien zulässig.

Es gibt keine Längenbeschränkung, allerdings trägt diese Zeichenfolge zur Gesamtlänge der Richtlinie bei, die wiederum beschränkt ist.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

Bietet eine Möglichkeit, Informationen zu einzelnen Anweisungen bereitzustellen. In IAM-Richtlinien sind für den Wert `Sid` nur die grundlegenden alphanummerischen Zeichen (A-Z, a-z und 0-9). Für andere AWS -Services, die ressourcenbasierte Richtlinien unterstützen, können für den Wert `Sid` andere Beschränkungen gelten. Bei einigen Diensten muss dieser Wert beispielsweise innerhalb eines eindeutig sein AWS-Konto, und bei einigen Diensten sind zusätzliche Zeichen wie Leerzeichen im `Sid` Wert zulässig.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

principal_id_string

Bietet eine Möglichkeit, einen Principal mithilfe des Amazon-Ressourcennamens (ARN) des IAM-Benutzers AWS-Konto, der IAM-Rolle, des Verbundbenutzers oder des Benutzers

mit angenommener Rolle anzugeben. Für eine AWS-Konto können Sie auch die Kurzform `AWS:accountnumber` anstelle des vollständigen ARN verwenden. Informationen zu sämtlichen Optionen einschließlich AWS -Services, übernommenen Rollen usw. finden Sie unter Angeben eines Auftraggebers.

Sie können das Sternchen (*) nur verwenden, um "alle/anonym" festzulegen. Es ist nicht möglich, damit einen Teil eines Namens oder ARN anzugeben.

`resource_string`

Besteht in den meisten Fällen aus einem Amazon-Ressourcennamen (ARN).

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::examplebucket/*"
```

`condition_type_string`

Gibt den zu testenden Bedingungstyp an, z. B. `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals` usw. Eine vollständige Liste der Bedingungstypen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingungsoperatoren.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Identifiziert den Bedingungsschlüssel, dessen Wert getestet wird, um festzustellen, ob die Bedingung erfüllt ist. AWS definiert eine Reihe von Bedingungsschlüsseln, die in allen AWS Diensten verfügbar sind`aws:PrincipalType`, einschließlich`aws:SecureTransport`, und`aws:userid`.

Eine Liste der AWS Bedingungsschlüssel finden Sie unterAWS Kontextschlüssel für globale Bedingungen. Informationen zu Bedingungsschlüsseln, die für einen Service spezifisch sind, finden Sie in der Dokumentation des betreffenden Service, wie z. B. in den folgenden Abschnitten:

- Angeben von Bedingungen in einer Richtlinie im Benutzerhandbuch für Amazon Simple Storage Service
- IAM-Richtlinien für Amazon EC2 im Amazon EC2 EC2-Benutzerhandbuch.

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Identifiziert den Wert von „condition_key_string“, der bestimmt, ob die Bedingung erfüllt ist. Eine vollständige Liste der gültigen Werte für einen Bedingungstyp finden Sie unter IAM-JSON-Richtlinienelemente: Bedingungsoperatoren.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
```

```
    "ID",
    "PostDateTime"
          ]
  }
}
```

AWS verwaltete Richtlinien für Jobfunktionen

Es wird empfohlen, Richtlinien zu verwenden, die die geringsten Rechte gewähren, d. h. nur die für die Durchführung einer Aufgabe erforderlichen Berechtigungen zu gewähren. Die sicherste Methode, die geringste Berechtigung zu erteilen, besteht darin, eine benutzerdefinierte Richtlinie mit nur den Berechtigungen zu schreiben, die Ihr Team benötigt. Sie müssen einen Prozess erstellen, damit Ihr Team bei Bedarf weitere Berechtigungen anfordern kann. Es erfordert Zeit und Fachwissen, um von Kunden verwaltete IAM-Richtlinien zu erstellen, die Ihrem Team nur die benötigten Berechtigungen bieten.

Um mit dem Hinzufügen von Berechtigungen für Ihre IAM-Identitäten (Benutzer, Benutzergruppen und Rollen) zu beginnen, können Sie Folgendes verwenden. AWS verwaltete Richtlinien AWS verwaltete Richtlinien decken allgemeine Anwendungsfälle ab und sind in Ihrem verfügbar. AWS-Konto AWS verwaltete Richtlinien gewähren keine Berechtigungen mit den geringsten Rechten. Sie müssen das Sicherheitsrisiko berücksichtigen, wenn Sie Ihren Auftraggeber mehr Berechtigungen gewähren, als sie für ihre Arbeit benötigen.

Sie können AWS verwaltete Richtlinien, einschließlich Jobfunktionen, an jede IAM-Identität anhängen. Um zu den Berechtigungen mit den geringsten Rechten zu wechseln, können Sie AWS Identity and Access Management Access Analyzer ausführen, um Prinzipale mit AWS verwalteten Richtlinien zu überwachen. Nachdem Sie erfahren haben, welche Berechtigungen sie verwenden, können Sie eine benutzerdefinierte Richtlinie schreiben oder eine Richtlinie mit nur den erforderlichen Berechtigungen für Ihr Team erstellen. Das ist weniger sicher, bietet aber mehr Flexibilität, wenn Sie erfahren, wie Ihr Team die Daten verwendet AWS.

AWS Die verwalteten Richtlinien für berufliche Funktionen sind so konzipiert, dass sie sich eng an den üblichen Aufgabenbereichen in der IT-Branche orientieren. Sie können mithilfe dieser Richtlinien ganz einfach die Berechtigungen erteilen, die Sie benötigen, um die Aufgaben, die von jemanden in einer bestimmten Auftragsfunktionen erwartet werden, auszuführen. Diese Richtlinien führen Berechtigungen für viele Services in einer einzigen Richtlinie zusammen, mit der sich leichter arbeiten lässt als mit auf vielen Richtlinien verteilten Berechtigungen.

Verwenden von Rollen zum Kombinieren von Services

In einigen Richtlinien werden IAM-Dienstrollen verwendet, damit Sie die Funktionen anderer AWS Dienste nutzen können. Diese Richtlinien gewähren Zugriff auf`iam:passrole`, was es einem Benutzer mit der Richtlinie ermöglicht, eine Rolle an einen AWS Dienst zu übergeben. Diese Rolle delegiert IAM-Berechtigungen an den AWS Dienst, damit dieser Aktionen in Ihrem Namen ausführen kann.

Erstellen Sie die Rollen entsprechend Ihren Anforderungen. Die Netzwerkadministrator-Richtlinie ermöglicht es beispielsweise einem Benutzer mit der Richtlinie, eine Rolle mit dem Namen „flow-logs-vpc“ an den Amazon-Service zu übergeben. CloudWatch CloudWatch verwendet diese Rolle, um den IP-Verkehr für vom Benutzer erstellte VPCs zu protokollieren und zu erfassen.

Damit die bewährten Sicherheitsmethoden befolgt werden, enthalten die Richtlinien für Auftragsfunktionen Filter, die die Namen der gültigen Rollen, die übergeben werden können, begrenzen. So vermeiden Sie, dass unnötige Berechtigungen zugewiesen werden. Wenn Ihre Benutzer die optionalen Servicerollen benötigen, müssen Sie eine Rolle erstellen, die der in der Richtlinie angegebenen Namenskonvention entspricht. Sie gewähren dann Berechtigungen für die Rolle. Sobald dies erledigt ist, kann der Benutzer den Service so konfigurieren, dass er die Rolle verwendet, wobei beliebige Berechtigungen der Rolle gewährt werden.

In den folgenden Abschnitten ist der Name jeder Richtlinie einen Link zur Richtliniendetailseite in der AWS Management Console. Dort können Sie das Richtliniendokument einsehen und die Berechtigungen überprüfen, die darin gewährt werden.

Auftragsfunktion

AWS Name der verwalteten Richtlinie: AdministratorAccess

Anwendungsfall: Dieser Benutzer hat vollständigen Zugriff und kann Berechtigungen an jeden Service und jede Ressource in AWS delegieren.

Richtlinienaktualisierungen: Diese Richtlinie wird AWS beibehalten und aktualisiert. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt alle Aktionen für alle AWS Dienste und für alle Ressourcen im Konto. Weitere Informationen zur verwalteten Richtlinie finden Sie AdministratorAccessim Referenzhandbuch für AWS verwaltete Richtlinien.

> **Note**
>
> Bevor ein IAM-Benutzer oder eine IAM-Rolle mit den in dieser Richtlinie festgelegten Berechtigungen auf die AWS Billing and Cost Management Konsole zugreifen kann, müssen Sie zuerst den IAM-Benutzer- und Rollenzugriff aktivieren. Befolgen Sie hierzu die Anweisungen in Schritt 1 des Tutorials zum Delegieren des Zugriffs auf die Abrechnungskonsole.

Fakturierung Auftragsfunktion

AWS Name der verwalteten Richtlinie: Abrechnung

Anwendungsfall: Dieser Benutzer muss Abrechnungsinformationen anzeigen und Zahlungen einrichten und autorisieren können. Der Benutzer kann die für den gesamten AWS Service anfallenden Kosten überwachen.

Aktualisierungen der Richtlinien: AWS Pflegt und aktualisiert diese Richtlinie. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Richtlinienbeschreibung: Diese Richtlinie gewährt volle Berechtigungen für die Verwaltung von Rechnungen, Kosten, Zahlungsmethoden, Budgets und Berichten. Weitere Beispiele für Kostenmanagement-Richtlinien finden Sie in den AWS Billing Richtlinienbeispielen im AWS Billing and Cost Management Benutzerhandbuch. Weitere Informationen zur verwalteten Richtlinie finden Sie unter Abrechnung im Referenzhandbuch für AWS verwaltete Richtlinien.

> **Note**
>
> Bevor ein IAM-Benutzer oder eine IAM-Rolle mit den in dieser Richtlinie festgelegten Berechtigungen auf die AWS Billing and Cost Management Konsole zugreifen kann, müssen Sie zuerst den IAM-Benutzer- und Rollenzugriff aktivieren. Befolgen Sie hierzu die Anweisungen in Schritt 1 des Tutorials zum Delegieren des Zugriffs auf die Abrechnungskonsole.

Datenbankadministrator-Auftragsfunktion

AWS Name der verwalteten Richtlinie: DatabaseAdministrator

Anwendungsfall: Dieser Benutzer richtet Datenbanken in der AWS Cloud ein, konfiguriert und verwaltet sie.

Richtlinienaktualisierungen: AWS Verwaltet und aktualisiert diese Richtlinie. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Richtlinienbeschreibung: Diese Richtlinie erteilt Berechtigungen, um Datenbanken zu erstellen, zu konfigurieren und zu verwalten. Es beinhaltet den Zugriff auf AWS Datenbankdienste wie Amazon DynamoDB, Amazon Relational Database Service (RDS) und Amazon Redshift. Zeigen Sie die Richtlinie für die vollständige Liste der Datenbank-Services an, die von dieser Richtlinie unterstützt werden. Weitere Informationen zur verwalteten Richtlinie finden Sie DatabaseAdministratorim Referenzhandbuch für AWS verwaltete Richtlinien.

Diese Richtlinie für Jobfunktionen unterstützt die Möglichkeit, Rollen an AWS Dienste zu übergeben. Die Richtlinie gewährt die `iam:PassRole`-Aktion ausschließlich für die in der folgenden Tabelle genannten Rollen. Weitere Informationen finden Sie unter Erstellen von Rollen und Anfügen von Richtlinien (Konsole) an späterer Stelle in diesem Thema.

Optionale IAM-Servicerollen für die Auftragsfunktion "Datenbankadministrator"

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	Wählen Sie diese AWS verwaltete Richtlinie
Der Benutzer kann RDS-Datenbanken überwachen	rds-monitoring-role	Amazon RDS Role for Enhanced Monitoring	Rolle bei Amazon RDS EnhancedMonitoring
Erlauben AWS Lambda Sie die Überwachung Ihrer Datenbank und den Zugriff auf externe Datenbanken	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	Wählen Sie diese AWS verwaltete Richtlinie
Lambda das Hochladen von Dateien in Amazon S3 und in Amazon Redshift Cluster mit DynamoDB ermöglichen	lambda_exec_role	AWS Lambda	Erstellen einer neuen verwalteten Richtlinie, wie im AWS Big Data Blog definiert
Erlauben Sie Lambda-Funktionen, als Trigger für Ihre DynamoDB-Tabellen zu fungieren	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
Erlauben Sie Lambda-Funktionen den Zugriff auf Amazon RDS in einer VPC	lambda-vpc-execution-role	Erstellen einer Rolle mit einer Vertrauen srichtlinie, wie im AWS Lambda Developer Guide definiert	AWSLambda VPCAccessExecutionRole
Erlauben AWS Data Pipeline Sie den Zugriff auf Ihre AWS Ressourcen	DataPipelineDefaultRole	Erstellen einer Rolle mit einer Vertrauen srichtlinie, wie im AWS Data Pipeline Developer Guide definiert	In der AWS Data Pipeline Dokumentation sind die erforderlichen Berechtigungen für diesen Anwendungsfall aufgeführt. Weitere Informationen finden Sie unter IAM-Rollen AWS Data Pipeline

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	Wählen Sie diese AWS verwaltete Richtlinie
Anwendungen, die auf Amazon EC2-Instances ausgeführt werden, können auf Ihre AWS -Ressourcen zugreifen	DataPipelineDefaultResourceRolle	Erstellen einer Rolle mit einer Vertrauensrichtlinie, wie im AWS Data Pipeline Developer Guide definiert	Amazon EC2 RoleforData PipelineRole

Auftragsfunktion für Data Scientist

AWS Name der verwalteten Richtlinie: DataScientist

Anwendungsfall: Dieser Benutzer führt Hadoop-Aufträge und -Abfragen. Der Benutzer greift auch auf Informationen für Datenanalysen und Business Intelligence zu und analysiert diese.

Richtlinienaktualisierungen: Diese Richtlinie wird AWS beibehalten und aktualisiert. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt Berechtigungen zum Erstellen, Verwalten und Ausführen von Abfragen in einem Amazon EMR-Cluster sowie zur Durchführung von Datenanalysen mit Tools wie Amazon QuickSight. Die Richtlinie beinhaltet den Zugriff auf zusätzliche Data-Scientist-Services wie AWS Data Pipeline Amazon EC2, Amazon Kinesis, Amazon Machine Learning und. SageMaker Zeigen Sie die Richtlinie für die vollständige Liste der Data Scientist-Services an, die von dieser Richtlinie unterstützt werden. Weitere Informationen zur verwalteten Richtlinie finden Sie DataScientistim Referenzhandbuch für AWS verwaltete Richtlinien.

Diese Richtlinie für Jobfunktionen unterstützt die Möglichkeit, Rollen an AWS Dienste zu übergeben. Eine Anweisung ermöglicht die Übergabe beliebiger Rollen an SageMaker. Eine weitere Anweisung gewährt die `iam:PassRole`-Aktion ausschließlich für die in der folgenden Tabelle genannten Rollen. Weitere Informationen finden Sie unter Erstellen von Rollen und Anfügen von Richtlinien (Konsole) an späterer Stelle in diesem Thema.

Optionale IAM-Servicerollen für die Auftragsfunktion "Data Scientist"

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	AWS verwaltete Richtlinie zur Auswahl
Ermöglichen Sie Amazon EC2-Instances den Zugriff auf für Cluster geeignete Dienste und Ressourcen	EMR-EC2_ DefaultRole	Amazon EMR für EC2	AmazonElasticMapReduceforEC2-Rolle
Amazon EMR-Zugriff für den Zugriff auf den Amazon EC2-Service und die Ressourcen für Cluster erlauben	EMR_ DefaultRole	Amazon EMR	Amazon EMR v2 ServicePolicy
Kinesis Managed Service für Apache Flink den Zugriff auf Streaming-Datenquellen erlauben	kinesis-*	Erstellen einer Rolle mit einer Vertrauensrichtlinie, wie im AWS Big Data Blog definiert	Weitere Informationen finden Sie im AWS Big Data Blog. Er enthält vier mögliche Optionen, je nach Ihrem Anwendungsfall.
Erlauben Sie den Zugriff AWS Data Pipeline auf Ihre Ressourcen AWS	DataPipelineDefaultRole	Erstellen einer Rolle mit einer Vertrauensrichtlinie, wie im AWS Data Pipeline Developer Guide definiert	In der AWS Data Pipeline Dokumentation sind die erforderlichen Berechtigungen für diesen Anwendungsfall aufgeführt. Weitere Informationen finden Sie unter IAM-Rollen AWS Data Pipeline

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	AWS verwaltete Richtlinie zur Auswahl
Anwendungen, die auf Amazon EC2-Instances ausgeführt werden, können auf Ihre AWS -Ressourcen zugreifen	DataPipelineDefaultResourceRolle	Erstellen einer Rolle mit einer Vertrauensrichtlinie, wie im AWS Data Pipeline Developer Guide definiert	Amazon EC2 RoleforData PipelineRole

Auftragsfunktion Developer Power User

AWS Name der verwalteten Richtlinie: Access PowerUser

Anwendungsfall: Dieser Benutzer führt Aufgaben zur Anwendungsentwicklung durch und kann Ressourcen und Dienste erstellen und konfigurieren, die die AWS bewusste Anwendungsentwicklung unterstützen.

Richtlinienaktualisierungen: AWS Verwaltet und aktualisiert diese Richtlinie. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: In der ersten Aussage dieser Richtlinie wird das `NotAction`Element verwendet, um alle Aktionen für alle AWS Dienste und Ressourcen außer AWS Identity and Access Management AWS Organizations, und zuzulassen AWS Account Management. Die zweite Anweisung erteilt IAM-Berechtigungen zum Erstellen einer serviceverknüpften Rolle. Dies wird von einigen Services benötigt, die auf Ressourcen in einem anderen Service zugreifen müssen, z. B. ein Amazon S3-Bucket. Außerdem erhalten Organisationen die Berechtigung, Informationen über die Organisation des Benutzers anzuzeigen, einschließlich der E-Mail des Verwaltungskontos und der Organisationsbeschränkungen. Obwohl diese Richtlinie den Zugriff von IAM und Organisationen einschränkt, erlaubt sie dem Benutzer alle IAM Identity Center-Aktionen auszuführen, wenn IAM Identity Center aktiviert ist. Es gewährt außerdem Kontoverwaltungsberechtigungen, mit denen Sie sehen können, welche AWS Regionen für das Konto aktiviert oder deaktiviert sind.

Netzwerkadministrator-Auftragsfunktion

AWS Name der verwalteten Richtlinie: NetworkAdministrator

Anwendungsfall: Dieser Benutzer hat die Aufgabe, AWS Netzwerkressourcen einzurichten und zu verwalten.

Aktualisierungen der Richtlinie: AWS Verwaltet und aktualisiert diese Richtlinie. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt Berechtigungen zum Erstellen und Verwalten von Netzwerkressourcen in Auto Scaling, Amazon EC2, AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing AWS Elastic Beanstalk,, Amazon SNS CloudWatch, CloudWatch Logs, Amazon S3, IAM und Amazon Virtual Private Cloud. Weitere Informationen zur verwalteten Richtlinie finden Sie NetworkAdministratorim Referenzhandbuch für AWS verwaltete Richtlinien.

Diese Jobfunktion erfordert die Fähigkeit, Rollen an AWS Dienste zu übergeben. Die Richtlinie gewährt `iam:GetRole` und `iam:PassRole` ausschließlich für die in der folgenden Tabelle genannten Rollen. Weitere Informationen finden Sie unter Erstellen von Rollen und Anfügen von Richtlinien (Konsole) an späterer Stelle in diesem Thema.

Optionale IAM-Servicerollen für die Auftragsfunktion "Network Administrator"

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	AWS verwaltete Richtlinie zur Auswahl
Ermöglicht Amazon VPC, im Namen des Benutzers Logs in CloudWatch Logs zu erstellen und zu verwalten , um den IP-Verkehr zu überwachen, der in und aus Ihrer VPC ein- und ausgeht	flow-logs-*	Erstellen einer Rolle mit einer Vertrauensrichtlinie, wie im Amazon VPC User Guide definiert	Für diesen Anwendungsfall gibt es keine bestehende AWS verwaltete Richtlinie, aber in der Dokumentation sind die erforderlichen Berechtigungen aufgeführt. Siehe

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	AWS verwaltete Richtlinie zur Auswahl
			Amazon VPC-Leitfaden.

Schreibgeschützter Zugriff

AWS Name der verwalteten Richtlinie: ReadOnly Access

Anwendungsfall Dieser Benutzer benötigt Nur-Lese-Zugriff auf alle Ressourcen eines AWS-Konto.

Richtlinienaktualisierungen: AWS Verwaltet und aktualisiert diese Richtlinie. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt Berechtigungen zum Auflisten, Abrufen, Beschreiben und anderweitigen Anzeigen von Ressourcen und deren Attributen. Es enthält keine mutierenden Funktionen wie Erstellen oder Löschen. Diese Richtlinie beinhaltet den schreibgeschützten Zugriff auf sicherheitsrelevante AWS Dienste wie und. AWS Identity and Access Management AWS Billing and Cost Management Zeigen Sie die Richtlinie für die vollständige Liste der Datenbank-Services an, die von dieser Richtlinie unterstützt werden.

Auftragsfunktion Sicherheitsprüfer

AWS Name der verwalteten Richtlinie: SecurityAudit

Anwendungsfall: Dieser Benutzer überwacht Konten auf die Einhaltung der Sicherheitsanforderungen. Dieser Benutzer kann auf Protokolle und Ereignisse zugreifen, um potenzielle Sicherheitsverstöße oder mögliche bösartige Aktivitäten zu untersuchen.

Richtlinienaktualisierungen: Diese Richtlinie wird AWS beibehalten und aktualisiert. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt Berechtigungen zum Anzeigen von Konfigurationsdaten für viele AWS Dienste und zum Überprüfen ihrer Protokolle. Weitere

Informationen zur verwalteten Richtlinie finden Sie SecurityAuditim Referenzhandbuch für AWS verwaltete Richtlinien.

Auftragsfunktion Support Benutzer

AWS Name der verwalteten Richtlinie: SupportUser

Anwendungsfall: Dieser Benutzer kontaktiert den AWS Support, erstellt Supportfälle und sieht sich den Status vorhandener Fälle an.

Aktualisierungen der Richtlinie: Diese Richtlinie wird AWS beibehalten und aktualisiert. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt Berechtigungen zum Erstellen und Aktualisieren von AWS Support Fällen. Weitere Informationen zur verwalteten Richtlinie finden Sie SupportUserim Referenzhandbuch für AWS verwaltete Richtlinien.

Funktion des Systemadministrators

AWS Name der verwalteten Richtlinie: SystemAdministrator

Anwendungsfall: Dieser Benutzer richtet Ressourcen für Entwicklungsvorgänge ein und verwaltet diese.

Richtlinienaktualisierungen: Diese Richtlinie wird AWS beibehalten und aktualisiert. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt Berechtigungen zum Erstellen und Verwalten von Ressourcen für eine Vielzahl von AWS Diensten, darunter Amazon AWS CloudTrail, CloudWatch, AWS CodeCommit, AWS CodeDeploy, AWS Config AWS Directory Service, Amazon EC2,, AWS Identity and Access Management, AWS Key Management Service AWS Lambda, Amazon RDS, Route 53, Amazon S3, Amazon SES, Amazon SQS und Amazon AWS Trusted Advisor VPC. Weitere Informationen zur verwalteten Richtlinie finden Sie SystemAdministratorim Referenzhandbuch für AWS verwaltete Richtlinien.

Diese Jobfunktion erfordert die Fähigkeit, Rollen an AWS Dienste zu übergeben. Die Richtlinie gewährt `iam:GetRole` und `iam:PassRole` ausschließlich für die in der folgenden Tabelle genannten Rollen. Weitere Informationen finden Sie unter Erstellen von Rollen und Anfügen von Richtlinien (Konsole) an späterer Stelle in diesem Thema. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Optionale IAM-Servicerollen für die Auftragsfunktion "System Administrator"

Anwendungsfall	Rollenname (* ist ein Platzhalter)	Auszuwählender Servicerollentyp	AWS verwaltete Richtlinie zur Auswahl
Apps, die in EC2-Instances in einem Amazon ECS-Cluster ausgeführt werden, können auf Amazon ECS zuzugreifen	ecr-sysadmin-*	Amazon EC2-Rolle für EC2 Container Service	Amazon EC2 EC2-Rolle Container Servicefor
Ein Benutzer kann Datenbanken überwachen	rds-monitoring-role	Amazon RDS Role for Enhanced Monitoring	Rolle bei EnhancedMonitoring AmazonRDS
Erlauben Sie Apps, die in EC2-Instances ausgeführt werden, den Zugriff auf Ressourcen AWS .	ec2-sysadmin-*	Amazon EC2	Beispielrichtlinie für eine Rolle, die Zugriff auf einen S3-Bucket gewährt, wie im Amazon EC2 EC2-Benutzerhandbuch beschrieben; nach Bedarf anpassen
Erlauben Sie Lambda, DynamoDB-Streams zu lesen und in Logs zu schreiben CloudWatch	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

Auftragsfunktion für Benutzer mit Lesezugriff

AWS Name der verwalteten Richtlinie: Access ViewOnly

Anwendungsfall: Dieser Benutzer kann eine Liste mit AWS Ressourcen und grundlegenden Metadaten im Konto dienstübergreifend einsehen. Der Benutzer kann Ressourceninhalt oder Metadaten, die über das Kontingent hinausgehen, nicht lesen und keine Informationen für Ressourcen auflisten.

Aktualisierungen der Richtlinie: AWS Verwaltet und aktualisiert diese Richtlinie. Eine Historie der Änderungen für diese Richtlinie finden Sie in der IAM-Konsole, und wählen Sie dann die Versionen der Datenlinien-Registerkarte. Weitere Informationen zu Auftragsfunktions-Richtlinienaktualisierungen finden Sie unter Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen.

Beschreibung der Richtlinie: Diese Richtlinie gewährt `List*Describe*`,,`Get*`,`View*`, und `Lookup*` Zugriff auf Ressourcen für AWS Dienste. Informationen darüber, welche Aktionen diese Richtlinie für die einzelnen Dienste beinhaltet, finden Sie unter ViewOnlyAccess. Weitere Informationen zur verwalteten Richtlinie finden Sie unter ViewOnlyAccess im Referenzhandbuch für AWS verwaltete Richtlinien.

Aktualisierungen der AWS verwalteten Richtlinien für Jobfunktionen

Diese Richtlinien werden alle von den Diensten verwaltet AWS und auf dem neuesten Stand gehalten, sodass sie auch Unterstützung für neue Dienste und neue Funktionen bieten, sobald diese von den AWS Diensten hinzugefügt werden. Diese Richtlinien können nicht von den Kunden geändert werden. Sie können eine Kopie der Richtlinie erstellen und dann die Kopie ändern, aber diese Kopie wird nicht automatisch aktualisiert, wenn neue Dienste und API-Operationen AWS eingeführt werden.

Für eine Auftragsfunktionsrichtlinie können Sie den Versionsverlauf sowie die Uhrzeit und das Datum jedes Updates in der IAM-Konsole anzeigen. Verwenden Sie dazu die Links auf dieser Seite, um die Richtliniendetails anzuzeigen. Wählen Sie dann die Versionen der Datenlinien, um die Versionen anzuzeigen. Auf dieser Seite werden die letzten 25 Versionen einer Richtlinie angezeigt. Um alle Versionen einer Richtlinie anzuzeigen, rufen Sie den AWS CLI Befehl get-policy-version oder den API-Vorgang GetPolicyVersion auf.

> **Note**
>
> Sie können bis zu fünf Versionen einer vom Kunden verwalteten Richtlinie verwenden, wobei jedoch der vollständige Versionsverlauf der AWS verwalteten Richtlinien AWS beibehalten wird.

Erstellen von Rollen und Anfügen von Richtlinien (Konsole)

Einige der zuvor aufgeführten Richtlinien ermöglichen die Konfiguration von AWS Diensten mit Rollen, die es diesen Diensten ermöglichen, Operationen in Ihrem Namen auszuführen. Die Auftragsfunktionsrichtlinien geben entweder genaue Rollennamen an, die Sie verwenden müssen, oder fügen zumindest ein Präfix an, das den ersten Teil des Namens, der verwendet werden kann, angibt. Führen Sie die Schritte in den folgenden Verfahren aus, um eine dieser Rollen zu erstellen.

Um eine Rolle für eine AWS-Service (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 Sie für Service oder Anwendungsfall einen Service und dann den Anwendungsfall aus. Anwendungsfälle werden durch den Service definiert, damit die für den Service erforderliche Vertrauensrichtlinie enthalten ist.
5. Wählen Sie Weiter aus.
6. Bei Berechtigungsrichtlinien hängen die Optionen vom ausgewählten Anwendungsfall ab:
   - Wenn der Dienst die Berechtigungen für die Rolle definiert, können Sie keine Berechtigungsrichtlinien auswählen.
   - Wählen Sie aus einer begrenzten Anzahl von Berechtigungsrichtlinien aus.
   - Wählen Sie aus allen Berechtigungsrichtlinien aus.
   - Wählen Sie keine Berechtigungsrichtlinien aus, erstellen Sie die Richtlinien, nachdem die Rolle erstellt wurde, und fügen Sie die Richtlinien dann der Rolle hinzu.
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. Die Optionen für den Rollennamen hängen vom Dienst ab:

   - Wenn der Dienst den Rollennamen definiert, können Sie den Rollennamen nicht bearbeiten.
   - Wenn der Dienst ein Präfix für den Rollennamen definiert, können Sie ein optionales Suffix eingeben.
   - Wenn der Dienst den Rollennamen nicht definiert, können Sie der Rolle einen Namen geben.

   > **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) Fügen Sie Tags als Schlüssel-Wert-Paare hinzu, um die Rolle leichter zu identifizieren, zu organisieren oder nach ihr zu suchen. 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).

Beispiel 1: Konfigurieren eines Benutzers als Datenbankadministrator (Konsole)

Dieses Beispiel zeigt die Schritte zum Konfigurieren von Alice, einem IAM-Benutzer, als Datenbankadministrator. Verwenden Sie die Informationen in der ersten Zeile der Tabelle in diesem Abschnitt und erlauben Sie dem Benutzer, die Amazon RDS-Überwachung zu aktivieren. Sie hängen die DatabaseAdministratorRichtlinie an den IAM-Benutzer von Alice an, damit dieser die Amazon-Datenbankdienste verwalten kann. Diese Richtlinie erlaubt es Alice auch, eine Rolle namens `rds-monitoring-role` an den Amazon-RDS-Service zu übergeben, die es dem Service erlaubt, die Amazon-RDS-Datenbanken in ihrem Namen zu überwachen.

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 Policies (Richtlinien) aus, geben Sie **`database`** in das Suchfeld ein und drücken Sie die Eingabetaste.
3. Wählen Sie das Optionsfeld für die DatabaseAdministratorRichtlinie aus, klicken Sie auf Aktionen und dann auf Anhängen.
4. Wählen Sie in der Liste der Entitäten Alice und dann Attach policy (Richtlinie anfügen) aus. Alice kann jetzt AWS Datenbanken verwalten. Damit Alice diese Datenbanken jedoch überwachen kann, müssen Sie die Servicerolle konfigurieren.
5. Klicken Sie im Navigationsbereich der IAM-Konsole auf Roles und wählen Sie dann Create role.
6. Wählen Sie den Rollentyp AWS Service und dann Amazon RDS.
7. Wählen Sie den Anwendungsfall Amazon RDS-Rolle für die verbesserte Überwachung.
8. Amazon RDS definiert die Berechtigungen für Ihre Rolle. Wählen Sie Next: Review (Weiter: Prüfen), um fortzufahren.
9. Der Rollenname muss einer der Namen sein, die in der DatabaseAdministrator Richtlinie angegeben sind, die Alice jetzt hat. Eine davon ist **`rds-monitoring-role`**. Geben Sie das für den Rollennamen ein.
10. (Optional) Geben Sie im Feld Role description (Rollenbeschreibung) eine Beschreibung für die neue Rolle ein.
11. Wählen Sie nach der Überprüfung der Details Create role (Rolle erstellen).
12. Alice kann jetzt RDS Enhanced Monitoring im Abschnitt Monitoring der Amazon RDS-Konsole aktivieren. Zum Beispiel kann sie dies tun, wenn sie eine DB-Instance erstellt, eine Read Replica

erstellt oder eine DB-Instance ändert. Sie müssen den Rollennamen, den sie erstellt haben (rds-monitoring-role), in das Feld „Überwachungsrolle“ eingeben, wenn sie Enable Enhanced Monitoring auf Ja setzen.

Beispiel 2: Konfigurieren eines Benutzers als Netzwerkadministrator (Konsole)

Dieses Beispiel zeigt die Schritte zum Konfigurieren von Jorge, einem IAM-Benutzer, als Netzwerkadministrator. Es verwendet die Informationen in der Tabelle in diesem Abschnitt, um es Jorge zu erlauben, den IP-Verkehr zu und von einer VPC zu überwachen. Es ermöglicht Jorge auch, diese Informationen in den Protokollen unter Logs zu erfassen. CloudWatch Sie fügen die NetworkAdministratorRichtlinie dem IAM-Benutzer von Jorge zu, damit dieser Netzwerkressourcen konfigurieren AWS kann. Diese Richtlinie erlaubt es Jorge auch, eine Rolle, deren Name mit `flow-logs*` beginnt, an Amazon EC2 zu übergeben, wenn Sie ein Flussprotokoll erstellen. In diesem Szenario gibt es im Gegensatz zu Beispiel 1 keinen vordefinierten Servicerollentyp, sodass Sie einige Schritte anders ausführen müssen.

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 Policies (Richtlinien) aus, geben Sie **`network`** in das Suchfeld ein und drücken Sie die Eingabetaste.
3. Wählen Sie das Optionsfeld neben der NetworkAdministratorRichtlinie aus, wählen Sie Aktionen und dann Anhängen aus.
4. Aktivieren Sie in der Liste der Benutzer das Kontrollkästchen neben Jorge und wählen Sie dann Attach policy (Richtlinie anfügen). Jorge kann jetzt AWS Netzwerkressourcen verwalten. Um IP-Datenverkehr in Ihrer VPC überwachen zu können, müssen Sie die Servicerolle jedoch konfigurieren.
5. Da die Servicerolle, die Sie erstellen müssen, keine vordefinierte verwaltete Richtlinie hat, müssen Sie diese zuerst erstellen. Wählen Sie im Navigationsbereich Policies und dann Create policy.
6. Wählen Sie im Abschnitt Policy editor (Richtlinien-Editor) die Option JSON aus und kopieren Sie den Text aus dem folgenden JSON-Richtliniendokument. Fügen Sie den folgenden Text in das JSON-Eingabefeld ein.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

7. Beheben Sie alle Sicherheitswarnungen, Fehler oder allgemeinen Warnungen, die während der Richtlinien-Validierung erzeugt wurden, und wählen Sie dann Weiter.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Next (Weiter) wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Umstrukturierung einer Richtlinie.

8. Geben Sie auf der Seite Review and create (Überprüfen und erstellen) als Richtliniennamen **`vpc-flow-logs-policy-for-service-role`** ein. Überprüfen Sie die Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die durch Ihre Richtlinie erteilten Berechtigungen zu sehen, und wählen Sie dann zum Speichern Create policy (Richtlinie erstellen) aus.

   Die neue Richtlinie wird in der Liste der verwalteten Richtlinien angezeigt und ist bereit.
9. Klicken Sie im Navigationsbereich der IAM-Konsole auf Roles und wählen Sie dann Create role.
10. Wählen Sie den Rollentyp AWS Service und danach Amazon EC2.
11. Wählen Sie den Anwendungsfall Amazon EC2.
12. Wählen Sie auf der Seite „Berechtigungsrichtlinien anhängen“ die zuvor erstellte Richtlinie aus, vpc-flow-logs-policy- for-service-role, und klicken Sie dann auf Weiter: Überprüfen.
13. Der Rollenname muss gemäß der NetworkAdministrator Richtlinie, über die Jorge jetzt verfügt, zugelassen sein. Es ist jeder Name zulässig, der mit `flow-logs-` beginnt. Geben Sie in diesem Beispiel **`flow-logs-for-jorge`** als Role name (Rollennamen) ein.

14. (Optional) Geben Sie im Feld Role description (Rollenbeschreibung) eine Beschreibung für die neue Rolle ein.
15. Wählen Sie nach der Überprüfung der Details Create role (Rolle erstellen).
16. Jetzt können Sie die Vertrauensrichtlinie konfigurieren, die für dieses Szenario erforderlich ist. Wählen Sie auf der Seite Rollen die flow-logs-for-jorgeRolle aus (den Namen, nicht das Kontrollkästchen). Wählen Sie auf der Detailseite für Ihre neue Rolle die Registerkarte Trust relationships (Vertrauensbeziehungen) und anschließend Edit trust relationship (Vertrauensbeziehung bearbeiten).
17. Ändern Sie die Zeile "Service" in Folgendes, wobei Sie den Eintrag für `ec2.amazonaws.com` ersetzen:

```
        "Service": "vpc-flow-logs.amazonaws.com"
```

18. Jorge kann jetzt Flow-Protokolle für eine VPC oder ein Subnetz in der Amazon-EC2-Konsole erstellen. Wenn Sie das Flow-Protokoll erstellen, geben Sie die flow-logs-for-jorgeRolle an. Diese Rolle hat die Berechtigungen, das Protokoll zu erstellen und Daten hineinzuschreiben.

AWS Kontextschlüssel für globale Bedingungen

Wenn ein Principal eine Anfrage an stellt AWS, werden AWS die Anforderungsinformationen in einem Anfragekontext zusammengefasst. Sie können das `Condition`-Element einer JSON-Richtlinie verwenden, um Schlüssel im Anforderungskontext mit Schlüsselwerten zu vergleichen, die Sie in Ihrer Richtlinie angeben. Die Anforderungsinformationen werden aus verschiedenen Quellen bereitgestellt, einschließlich des Principals, der die Anfrage stellt, der Ressource, für die die Anfrage gestellt wird, und der Metadaten zur Anfrage selbst.

Globale Bedingungsschlüssel können für alle AWS Dienste verwendet werden. Diese Bedingungsschlüssel können zwar in allen Richtlinien verwendet werden, der Schlüssel ist jedoch nicht in jedem Anforderungskontext verfügbar. Beispielsweise ist der `aws:SourceAccount` Bedingungsschlüssel nur verfügbar, wenn der Aufruf Ihrer Ressource direkt von einem AWS Service Principal erfolgt. Weitere Informationen zu den Umständen, unter denen ein globaler Schlüssel im Anforderungskontext enthalten ist, finden Sie in den Verfügbarkeitsinformationen für jeden Schlüssel.

Einige einzelne Dienste erstellen ihre eigenen Bedingungsschlüssel, die im Anforderungskontext für andere Dienste verfügbar sind. Serviceübergreifende Bedingungsschlüssel sind eine Art globaler Bedingungsschlüssel, die ein Präfix enthalten, das dem Namen des Dienstes entspricht, z. B. `ec2:` oder`lambda:`, aber für andere Dienste verfügbar sind.

Dienstspezifische Bedingungsschlüssel werden für die Verwendung mit einem einzelnen AWS Dienst definiert. Mit Amazon S3 können Sie beispielsweise eine Richtlinie mit dem `s3:VersionId` Bedingungsschlüssel schreiben, um den Zugriff auf eine bestimmte Version eines Amazon S3 S3-Objekts zu beschränken. Dieser Bedingungsschlüssel ist für den Service einzigartig, was bedeutet, dass er nur mit Anfragen an den Amazon S3 S3-Service funktioniert. Informationen zu servicespezifischen Bedingungsschlüsseln finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Wählen Sie dort den Service aus, dessen Schlüssel Sie anzeigen möchten.

Note

Wenn Sie Bedingungsschlüssel verwenden, die nur unter bestimmten Umständen verfügbar sind, können Sie die IfExistsVersionen der Bedingungsoperatoren verwenden. Wenn die entsprechenden Bedingungsschlüssel im Kontext einer Anforderung fehlen, kann die Richtlinienauswertung fehlschlagen. Verwenden Sie beispielsweise den folgenden Bedingungsblock mit `...IfExists`-Operatoren, um abzugleichen, ob eine Anforderung aus einem bestimmten IP-Bereich oder von einer bestimmten VPC stammt. Wenn einer oder beide Schlüssel nicht im Anforderungskontext enthalten sind, gibt die Bedingung weiterhin `true` zurück. Die Werte werden nur überprüft, wenn der angegebene Schlüssel im Anforderungskontext enthalten ist. Weitere Informationen darüber, wie eine Richtlinie bewertet wird, wenn ein Schlüssel für andere Operatoren nicht vorhanden ist, finden Sie unter Bedingungsoperatoren.

```
"Condition": {
    "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
    "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
}
```

Important

Um Ihre Bedingung mit einem Anforderungskontext mit mehreren Schlüsselwerten zu vergleichen, müssen Sie die Set-Operatoren `ForAllValues` oder `ForAnyValue` verwenden. Verwenden Sie Satz-Operatoren nur mit mehrwertigen Bedingungsschlüssel. Verwenden Sie keine Satz-Operatoren mit einzelwertigen Bedingungsschlüssel. Weitere Informationen finden Sie unter Mehrwertige Kontextschlüssel.

Eigenschaften des Prinzipals	Eigenschaften einer Rollensitzung	Eigenschaften des Netzwerks	Eigenschaften der Ressource	Eigenschaften der Anfrage
war: PrincipalArn als: Principal Account aws: Principal Org Pfade aws: Principal Org ID PrincipalTagaws/tag-key als: PrincipalIs AWSService aws: Principal Service Name aws: Principal Service NamesList als: Principal Type aws:userid aws:username	als: Federated Provider aws: TokenIssue Zeit als: MultiFactor AuthAge als: MultiFactor AuthPresent AWSInstan ceSource: EC2 Vpc AWS: EC2 PrivateIPv4 InstanceSource als: SourceIde ntity ec2: RoleDeliv ery ec2: Arn SourceInstance Klebstoff: RoleAssumed Von	aws: SourceIp aws: SourceVpc als: SourceVpce aws: VpcSource Ip	aws: ResourceA ccount aws: ResourceO rg Pfade aws: ResourceO rg ID aws:Resou rceTag/Tag-Taste	aws: CalledVia aws: CalledVia Zuerst aws: CalledVia Zuletzt AWS: via AWSService war: CurrentTi me als: EpochTime aws:referer als: Requested Region RequestTagaws/Tag-Schlüssel als: TagKeys aws: SecureTra nsport war: SourceArn aws: SourceAcc ount aws: SourceOrg Pfade

Eigenschaften des Prinzipals	Eigenschaften einer Rollensitzung	Eigenschaften des Netzwerks	Eigenschaften der Ressource	Eigenschaften der Anfrage
	Klebstoff: CredentialIssuingService Lambda: ArnSourceFunction ssm: ArnSourceInstance Identitätsspeicher: UserId			aws: SourceOrgID aws: UserAgent

Eigenschaften des Prinzipals

Verwenden Sie die folgenden Bedingungsschlüssel, um Details über den Prinzipal, der die Anfrage stellt, mit den Prinzipaleigenschaften zu vergleichen, die Sie in der Richtlinie angeben. Eine Liste der Prinzipale, die Anfragen stellen können, finden Sie unterAngeben eines Auftraggebers.

Inhalt

- war: PrincipalArn
- als: PrincipalAccount
- aws: PrincipalOrg Pfade
- aws: PrincipalOrg ID
- PrincipalTagaws/tag-key
- als: PrincipalIs AWSService
- aws: PrincipalService Name
- aws: PrincipalService NamesList
- als: PrincipalType
- aws:userid

- aws:username

war: PrincipalArn

Verwenden Sie diesen Schlüssel, um den Amazon-Ressourcennamen (ARN) des Auftraggebers, von dem die Anforderung stammt, mit dem ARN zu vergleichen, den Sie in der Richtlinie angeben. Bei IAM-Rollen gibt der Anforderungskontext den ARN der Rolle zurück, nicht den ARN des Benutzers, der die Rolle übernommen hat.

- Verfügbarkeit – Dieser Schlüssel ist im Anforderungskontext aller signierten Anforderungen enthalten. Anonyme Anforderungen enthalten diesen Schlüssel nicht. In diesem Bedingungsschlüssel können Sie die folgenden Arten von Prinzipals angeben:
  - IAM-Rolle
  - IAM-Benutzer
  - AWS STS föderierte Benutzersitzung
  - AWS-Konto Root-Benutzer
- Datentyp — ARN, Zeichenfolge

  AWS empfiehlt, beim Vergleich von ARNs ARN-Operatoren anstelle von Zeichenkettenoperatoren zu verwenden.
- Werttyp - Einzelwertig
- Beispielwerte Die folgende Liste zeigt den Wert des Anforderungskontextes, der für verschiedene Typen von Prinzipalen zurückgegeben wird, die Sie im `aws:PrincipalArn` Bedingungsschlüssel angeben können:
  - IAM-Rolle – Der Anforderungskontext enthält den folgenden Wert für den Bedingungsschlüssel `aws:PrincipalArn`. Geben Sie den ARN der angenommenen Rollensitzung nicht als Wert für diesen Bedingungsschlüssel an. Weitere Informationen über die angenommene Rollen des Sitzungs-Prinzipals finden Sie unter Rollensitzungsgsprinzipale.

    ```
    arn:aws:iam::123456789012:role/role-name
    ```
  - IAM-Benutzer – Der Anforderungskontext enthält den folgenden Wert für den Bedingungsschlüssel `aws:PrincipalArn`.

    ```
    arn:aws:iam::123456789012:user/user-name
    ```

- AWS STS Verbundbenutzersitzungen — Der Anforderungskontext enthält den folgenden Wert für den Bedingungsschlüssel. `aws:PrincipalArn`

  ```
  arn:aws:sts::123456789012:federated-user/user-name
  ```

- AWS-Konto Root-Benutzer — Der Anforderungskontext enthält den folgenden Wert für den Bedingungsschlüssel`aws:PrincipalArn`. Wenn Sie den Root-Benutzer-ARN als Wert für den Bedingungsschlüssel `aws:PrincipalArn` angeben, werden die Berechtigungen nur für den Root-Benutzer des AWS-Konto eingeschränkt. Dies unterscheidet sich von der Angabe des Root-Benutzer-ARN im Prinzipal-Element einer ressourcenbasierten Richtlinie, die die Autorität an die AWS-Konto delegiert. Weitere Informationen zur Angabe des Root-Benutzer-ARN im Prinzipal-Element einer ressourcenbasierten Richtlinie finden Sie unter AWS-Konto Schulleiter.

  ```
  arn:aws:iam::123456789012:root
  ```

Sie können den Root-Benutzer-ARN als Wert für den Bedingungsschlüssel `aws:PrincipalArn` in AWS Organizations Service Control Policies (SCPs) angeben. SCPs sind eine Art von Organisationsrichtlinien, die zum Verwalten von Berechtigungen in Ihrer Organisation verwendet werden und sich nur auf Mitgliedskonten in der Organisation auswirken. Ein SCP beschränkt die Berechtigungen für IAM-Benutzer und -Rollen in Mitgliedskonten, einschließlich des Stammverzeichnisses des Mitgliedskonten. Weitere Informationen zu den Auswirkungen von SCPs auf Berechtigungen finden Sie unter SCP-Auswirkungen auf Berechtigungen im Benutzerhandbuch für Organisationen.

als: PrincipalAccount

Verwenden Sie diesen Schlüssel, um das Konto, zu dem der anfordernde Auftraggeber gehört, mit der Konto-ID zu vergleichen, die Sie in der Richtlinie angeben. Bei anonymen Anforderungen gibt der Anforderungskontext `anonymous` zurück.

- Verfügbarkeit – Dieser Schlüssel wird in den Anforderungskontext für alle Anforderungen, einschließlich anonymer Anforderungen, aufgenommen.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Im folgenden Beispiel wird der Zugriff verweigert, mit Ausnahme von Auftraggebern mit der Kontonummer `123456789012`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:accountID:resource"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws: PrincipalOrg Pfade

Verwenden Sie diesen Schlüssel, um den AWS Organizations Pfad für den Prinzipal, der die Anfrage stellt, mit dem Pfad in der Richtlinie zu vergleichen. Dieser Prinzipal kann ein IAM-Benutzer, eine IAM-Rolle, ein Verbundbenutzer oder sein. Root-Benutzer des AWS-Kontos In einer Richtlinie stellt dieser Bedingungsschlüssel sicher, dass der Anforderer ein Kontomitglied innerhalb des angegebenen Organisationsstammes oder der Organisationseinheiten in AWS Organizations ist. Ein AWS Organizations Pfad ist eine Textdarstellung der Struktur einer Organisationseinheit. Weitere Hinweise zum Verwenden und Verstehen von Pfaden finden Sie unter Der AWS Organizations - Entitätspfad.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Auftraggeber Mitglied einer Organisation ist. Anonyme Anforderungen enthalten diesen Schlüssel nicht.
- Datentyp — Zeichenfolge (Liste)
- Werttyp - Mehrwertig

> **Note**
>
> Organisations-IDs sind global eindeutig, Organisationeinheiten-IDs und Stamm-IDs sind jedoch nur innerhalb einer Organisation eindeutig. Dies bedeutet, dass keine zwei Organisationen dieselbe Organisations-ID verwenden. Eine andere Organisation verfügt jedoch möglicherweise über eine Organisationseinheit oder ein Stammverzeichnis mit derselben ID wie Ihre. Es wird empfohlen, immer die Organisations-ID anzugeben, wenn Sie eine Organisationseinheit oder ein Stammverzeichnis angeben.

Beispielsweise gibt die folgende Bedingung `true` für Auftraggeber in Konten zurück, die direkt an die `ou-ab12-22222222`-Organisationseinheit angefügt sind, aber nicht an die untergeordneten Organisationseinheiten.

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

Die folgende Bedingung gibt `true` für Auftraggeber in einem Konto zurück, das direkt an die Organisationseinheit oder eine ihrer untergeordneten Organisationseinheiten angefügt ist. Wenn Sie einen Platzhalter hinzufügen, müssen Sie den `StringLike`-Bedingungsoperator verwenden.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

Die folgende Bedingung gibt `true` für Auftraggeber in einem Konto zurück, das direkt an die Organisationseinheit oder eine ihrer untergeordneten Organisationseinheiten angefügt ist. Die vorherige Bedingung gilt für die Organisationseinheit oder für untergeordnete Elemente. Die folgende Bedingung gilt nur für untergeordnete Elemente (und für untergeordnete Elemente dieser Kinder).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

Die folgende Bedingung ermöglicht den Zugriff für jeden Auftraggeber in der `o-a1b2c3d4e5`-Organisation, unabhängig von der übergeordneten Organisationseinheit.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` ist ein mehrwertiger Bedingungsschlüssel. Mehrwertige Bedingungsschlüssel können im Anforderungskontext mehrere Werte haben. Wenn Sie mehrere Werte mit dem `ForAnyValue`-Bedingungsoperator verwenden, muss der Pfad des Auftraggebers mit einem der in der Richtlinie aufgeführten Pfade übereinstimmen. Weitere Hinweise zu mehrwertigen Bedingungsschlüsseln finden Sie unter Mehrwertige Kontextschlüssel.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws: PrincipalOrg ID

Verwenden Sie diesen Schlüssel, um die ID der Organisation, AWS Organizations zu der der anfordernde Principal gehört, mit der in der Richtlinie angegebenen ID zu vergleichen.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Auftraggeber Mitglied einer Organisation ist. Anonyme Anforderungen enthalten diesen Schlüssel nicht.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Dieser globale Schlüssel stellt eine Alternative zum Auflisten aller Konto-IDs für alle AWS -Konten in einer Organisation dar. Sie können diesen Bedingungsschlüssel verwenden, um das Angeben des Elements `Principal` in einer resource-based-Richtlinie zu vereinfachen. Sie können die Organisations-ID im Bedingungselement angeben. Wenn Sie Konten hinzufügen und entfernen, schließen Richtlinien, die den `aws:PrincipalOrgID`-Schlüssel enthalten, automatisch die richtigen Konten ein und müssen nicht manuell aktualisiert werden.

Mit der folgenden Amazon S3-Bucket-Richtlinie können beispielsweise Mitglieder aller Konten in der Organisation `o-xxxxxxxxxxx` ein Objekt in den Bucket `policy-ninja-dev` einfügen.

```
 {
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AllowPutObject",
    "Effect": "Allow",
    "Principal": "*",
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::policy-ninja-dev/*",
    "Condition": {"StringEquals":
      {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
    }
  }
}
```

> **Note**
>
> Diese globale Bedingung gilt auch für das Masterkonto einer AWS -Organisation. Diese Richtlinie verhindert, dass alle Hauptbenutzer außerhalb der angegebenen Organisation auf den Amazon-S3-Bucket zugreifen können. Dazu gehören alle AWS Dienste, die mit Ihren internen Ressourcen interagieren, z. B. das AWS CloudTrail Senden von Protokolldaten an Ihre Amazon S3 S3-Buckets. Informationen dazu, wie Sie auf sichere Weise Zugriff auf AWS Dienste gewähren können, finden Sie unterals: PrincipalIs AWSService.

Weitere Informationen zu AWS Organizations finden Sie unter Was ist AWS Organizations? im AWS Organizations Benutzerhandbuch.

PrincipalTagaws/tag-key

Verwenden Sie diesen Schlüssel, um das Tag, das dem Auftraggeber angefügt ist, der die Anforderung stellt, mit dem Tag zu vergleichen, das Sie in der Richtlinie angeben. Wenn dem Auftraggeber mehr als ein Tag angefügt ist, enthält der Anforderungskontext einen `aws:PrincipalTag`-Schlüssel für jeden angefügten Tag-Schlüssel.

- Availability (Verfügbarkeit) – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Auftraggeber ein IAM-Benutzer mit angefügten Tags ist. Er ist für einen Auftraggeber enthalten,

der eine IAM-Rolle mit angefügten Tags oder Sitzungs-Tags verwendet. Anonyme Anforderungen enthalten diesen Schlüssel nicht.

- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Sie können einem Benutzer oder einer Rolle benutzerdefinierte Attribute in Form eines Schlüssel-Wert-Paares hinzufügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen. Sie können `aws:PrincipalTag` für die Zugriffskontrolle für AWS -Auftraggeber einsetzen.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen können, die es Benutzern mit dem Tag **`department=hr`** erlaubt, IAM-Benutzer, -Gruppen oder -Rollen zu verwalten. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

als: PrincipalIs AWSService

Verwenden Sie diesen Schlüssel, um zu überprüfen, ob der Aufruf Ihrer Ressource direkt von einem AWS Service Principal erfolgt. Zum Beispiel verwendet AWS CloudTrail den Service-Prinzipal `cloudtrail.amazonaws.com`, um Protokolle in Ihr Amazon S3-Bucket zu schreiben. Der Anforderungskontextschlüssel wird auf true festgelegt, wenn ein Dienst einen Dienstauftraggeber verwendet, um eine direkte Aktion für Ihre Ressourcen auszuführen. Der Kontextschlüssel wird

auf false gesetzt, wenn der Dienst die Anmeldeinformationen eines IAM-Auftraggeber verwendet, um eine Anforderung im Namen des Auftraggeber zu stellen. Sie wird auch verweigert, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen.

- Verfügbarkeit – Dieser Schlüssel ist im Anforderungskontext aller signierten API-Anforderungen vorhanden, die AWS -Anmeldeinformationen. Anonyme Anforderungen enthalten diesen Schlüssel nicht.
- Datentyp — Boolean
- Werttyp - Einzelwertig

Sie können diesen Bedingungsschlüssel verwenden, um den Zugriff auf Ihre vertrauenswürdigen Identitäten und erwarteten Netzwerkadressen zu beschränken und gleichzeitig den Zugriff auf Dienste sicher zu gewähren. AWS

Im folgenden Beispiel für eine Amazon S3 S3-Bucket-Richtlinie ist der Zugriff auf den Bucket eingeschränkt, es sei denn, die Anfrage stammt von einem Service Principal `vpc-111bbb22` oder stammt von einem Service Principal, z. B. CloudTrail

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

Im folgenden Video erfahren Sie mehr darüber, wie Sie den Bedingungsschlüssel `aws:PrincipalIsAWSService` in einer Richtlinie verwenden können.

Gewähren Sie Ihren autorisierten Benutzern, erwarteten Netzwerkstandorten und AWS Diensten gemeinsam auf sichere Weise Zugriff.

aws: PrincipalService Name

Verwenden Sie diesen Schlüssel, um den Dienstauftraggeber-Namen in der Richtlinie mit dem Dienstauftraggeber zu vergleichen, der Anforderungen an Ihre Ressourcen stellt. Sie können diesen Schlüssel verwenden, um zu überprüfen, ob dieser Aufruf von einem bestimmten Dienstauftraggeber erfolgt. Wenn ein Dienstauftraggeber eine direkte Anforderung an Ihre Ressource stellt, enthält der `aws:PrincipalServiceName`-Schlüssel den Namen des Dienstauftraggebers. Der AWS CloudTrail Dienstprinzipalname lautet beispielsweise`cloudtrail.amazonaws.com`.

- Verfügbarkeit — Dieser Schlüssel ist in der Anfrage enthalten, wenn der Anruf von einem AWS Dienstprinzipal getätigt wird. Dieser Schlüssel ist in keiner anderen Situation vorhanden, einschließlich der folgenden:
  - Sie wird auch verweigert, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen.
  - Wenn der Dienst die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um eine Anforderung im Namen des Auftraggebers zu stellen.
  - Wenn der Anruf direkt von einem IAM-Auftraggeber getätigt wird.
  - Wenn der Anruf von einem anonymen Antragsteller getätigt wird.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Sie können diesen Bedingungsschlüssel verwenden, um den Zugriff auf Ihre vertrauenswürdigen Identitäten und erwarteten Netzwerkadressen zu beschränken und gleichzeitig den Zugriff auf einen AWS Dienst sicher zu gewähren.

Im folgenden Beispiel für eine Amazon S3 S3-Bucket-Richtlinie ist der Zugriff auf den Bucket eingeschränkt, es sei denn, die Anfrage stammt von einem Service Principal `vpc-111bbb22` oder stammt von einem Service Principal, z. B. CloudTrail

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
}
```

aws: PrincipalService NamesList

Dieser Schlüssel enthält eine Liste aller-DienstauftraggeberNamen, die zum Dienst gehören. Dies ist ein erweiterter Bedingungsschlüssel. Sie können damit den Service daran hindern, nur von einer bestimmten Region aus auf Ihre Ressource zuzugreifen. Einige Dienste können regionale Dienstauftraggeber erstellen, um eine bestimmte Instance des Dienstes innerhalb einer bestimmten Region anzugeben. Sie können den Zugriff auf eine Ressource auf eine bestimmte Instance des Dienstes beschränken. Wenn ein Dienstauftraggeber eine direkte Anforderung an Ihre Ressource stellt, enthält der `aws:PrincipalServiceNamesList` eine ungeordnete Liste aller Dienst-Auftraggebernamen, die mit der regionalen Instance des Dienstes verbunden sind.

- Verfügbarkeit — Dieser Schlüssel ist in der Anfrage enthalten, wenn der Anruf von einem AWS Service Principal getätigt wird. Dieser Schlüssel ist in keiner anderen Situation vorhanden, einschließlich der folgenden:
  - Sie wird auch verweigert, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen.
  - Wenn der Dienst die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um eine Anforderung im Namen des Auftraggebers zu stellen.
  - Wenn der Anruf direkt von einem IAM-Auftraggeber getätigt wird.
  - Wenn der Anruf von einem anonymen Antragsteller getätigt wird.
- Datentyp — Zeichenfolge (Liste)
- Werttyp - Mehrwertig

`aws:PrincipalServiceNamesList` ist ein mehrwertiger Bedingungsschlüssel. Mehrwertige Bedingungsschlüssel können im Anforderungskontext mehrere Werte haben. Sie müssen die Set-Operatoren `ForAnyValue` oder `ForAllValues` zusammen mit String-Bedingungsoperatoren für diesen Schlüssel verwenden. Weitere Hinweise zu mehrwertigen Bedingungsschlüsseln finden Sie unter Mehrwertige Kontextschlüssel.

als: PrincipalType

Verwenden Sie diesen Schlüssel, um den Typ des Auftraggebers, der die Anforderung stellt, mit dem Auftraggebertyp zu vergleichen, den Sie in der Richtlinie angeben. Weitere Informationen finden Sie unter Angeben eines Auftraggebers. Für konkrete Beispiele von `principal`-Schlüsselwerten, siehe Auftraggeber-Schlüsselwerte.

- Verfügbarkeit – Dieser Schlüssel ist im Anforderungskontext für alle Anforderungen, einschließlich anonymer Anforderungen, enthalten.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

aws:userid

Verwenden Sie diesen Schlüssel, um die Auftraggeber-ID des Anforderers mit der ID zu vergleichen, die Sie in der Richtlinie angeben. Bei IAM-Benutzern ist der Anforderungskontextwert die Benutzer-ID. Bei IAM-Rollen kann dieses Werteformat variieren. Weitere Informationen dazu, wie die Informationen für verschiedene Auftraggeber angezeigt werden, finden Sie unter Angeben eines Auftraggebers. Für konkrete Beispiele von `principal`-Schlüsselwerten, siehe Auftraggeber-Schlüsselwerte.

- Verfügbarkeit – Dieser Schlüssel ist im Anforderungskontext für alle Anforderungen, einschließlich anonymer Anforderungen, enthalten.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

aws:username

Verwenden Sie diesen Schlüssel, um den Benutzernamen des Anforderers mit dem Benutzernamen zu vergleichen, den Sie in der Richtlinie angeben. Weitere Informationen dazu, wie die Informationen

für verschiedene Auftraggeber angezeigt werden, finden Sie unter Angeben eines Auftraggebers. Für konkrete Beispiele von `principal`-Schlüsselwerten, siehe Auftraggeber-Schlüsselwerte.

- Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext für IAM-Benutzer enthalten. Anonyme Anfragen und Anfragen, die mithilfe der Rollen Root-Benutzer des AWS-Kontos oder IAM gestellt werden, enthalten diesen Schlüssel nicht. Anforderungen, die mit IAM Identity Center-Anmeldeinformationen gestellt werden, enthalten diesen Schlüssel nicht im Kontext.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Eigenschaften einer Rollensitzung

Verwenden Sie die folgenden Bedingungsschlüssel, um die Eigenschaften der Rollensitzung zum Zeitpunkt der Sitzungsgenerierung zu vergleichen. Diese Bedingungsschlüssel sind nur verfügbar, wenn eine Anfrage von einem Principal mit Rollensitzungs- oder Verbundbenutzeranmeldedaten gestellt wird. Die Werte für diese Bedingungsschlüssel sind in das Sitzungstoken der Rolle eingebettet.

Eine Rolle ist eine Art von Principal. Sie können auch die Bedingungsschlüssel aus dem Eigenschaften des Prinzipals Abschnitt verwenden, um die Eigenschaften einer Rolle auszuwerten, wenn eine Rolle eine Anfrage stellt.

Inhalt

- ssm: Arn SourceInstance
- Identitätsspeicher: UserId

als: FederatedProvider

Verwenden Sie diesen Schlüssel, um die Auftraggeber-ID des Anforderers (IdP) mit der ID zu vergleichen, die Sie in der Richtlinie angeben. Dies bedeutet, dass mithilfe des `AssumeRoleWithWebIdentity` AWS STS Vorgangs eine IAM-Rolle übernommen wurde. Wenn die temporären Anmeldeinformationen der resultierenden Rollensitzung verwendet werden, um eine Anforderung zu stellen, identifiziert der Anforderungskontext den IdP, der die ursprüngliche Verbundidentität authentifiziert hat.

- Verfügbarkeit – Dieser Schlüssel ist vorhanden, wenn der Prinzipal ein Rollensitzungsprinzipal ist und diese Sitzung ausgegeben wurde, als eine Rolle mit `AssumeRoleWithWebIdentity` übernommen wurde.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Wenn der Benutzer beispielsweise über Amazon Cognito authentifiziert wurde, enthält der Anforderungskontext den Wert `cognito-identity.amazonaws.com`. Wenn der Benutzer über Login with Amazon authentifiziert wurde, enthält der Anforderungskontext den Wert `www.amazon.com`.

Sie können jeden einwertigen Bedingungsschlüssel als Variable verwenden. Im folgenden Beispiel verwendet die ressourcenbasierte Richtlinie `aws:FederatedProvider`-Schlüssel als Richtlinienvariable im ARN einer Ressource. Diese Richtlinie ermöglicht es Benutzern, die sich über einen Identitätsanbieter authentifiziert haben, Objekte aus einem Amazon-S3-Bucket mit einem für den ausstellenden Identitätsanbieter spezifischen Pfad abzurufen.

aws: TokenIssue Zeit

Verwenden Sie diesen Schlüssel, um das Datum und die Uhrzeit der Ausstellung der Sicherheitsanmeldeinformationen mit dem Datum und der Uhrzeit zu vergleichen, das bzw. die Sie in der Richtlinie angeben.

- Availability – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Auftraggeber temporäre Anmeldeinformationen für die Anforderung verwendet. Der Schlüssel ist

weder in AWS API AWS CLI- noch in AWS SDK-Anfragen enthalten, die mit Zugriffsschlüsseln gestellt werden.

- Datentyp — Datum
- Werttyp - Einzelwertig

Informationen darüber, welche Services die Verwendung von temporären Anmeldeinformationen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

als: MultiFactor AuthAge

Verwenden Sie diesen Schlüssel, um die Anzahl der Sekunden, die seit dem Zeitpunkt, zu dem der anfordernde Auftraggeber per MFA autorisiert wurde, verstrichen sind mit der Anzahl, die Sie in der Richtlinie angeben, zu vergleichen. Weitere Informationen zu MFA finden Sie unter Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS.

> **Important**
>
> Dieser Bedingungsschlüssel ist nicht für föderierte Identitäten oder Anfragen vorhanden, die mithilfe von Zugriffsschlüsseln zum Signieren von AWS CLI-, AWS API- oder AWS SDK-Anfragen gestellt wurden. Weitere Informationen zum Hinzufügen von MFA-Schutz zu API-Vorgängen mit temporären Sicherheitsanmeldedaten finden Sie unterKonfigurieren eines MFA-geschützten API-Zugriffs.
> Um zu überprüfen, ob MFA zur Validierung von IAM-Verbundidentitäten verwendet wird, können Sie die Authentifizierungsmethode von Ihrem Identitätsanbieter AWS als Sitzungs-Tag an übergeben. Details hierzu finden Sie unter Sitzungs-Tags übergeben AWS STS. Um MFA für IAM Identity Center-Identitäten durchzusetzen, können Sie Attribute für die Zugriffskontrolle aktivieren, um einen SAML-Assertion-Anspruch mit der Authentifizierungsmethode von Ihrem Identitätsanbieter an IAM Identity Center weiterzuleiten.

- Verfügbarkeit — Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Principal temporäre Sicherheitsanmeldedaten verwendet, um die Anfrage zu stellen. Policen mit MFA-Bedingungen können beigefügt werden an:
  - Einem IAM-Benutzer oder einer IAM-Gruppe
  - Einer Ressource, wie zum Beispiel ein Amazon S3-Bucket, eine Amazon SQS-Warteschlange oder ein Amazon SNS-Thema
  - Die Vertrauensrichtlinie einer IAM-Rolle, die von einem Benutzer übernommen werden kann

- Datentyp — Numerisch
- Werttyp - Einzelwertig

als: MultiFactor AuthPresent

Verwenden Sie diesen Schlüssel, um zu überprüfen, ob die Multi-Faktor-Authentifizierung (MFA) verwendet wurde, um die temporären Sicherheitsanmeldeinformationen zu validieren, die die Anfrage gestellt haben.

> **Important**
>
> Dieser Bedingungsschlüssel ist nicht für föderierte Identitäten oder Anfragen vorhanden, die mithilfe von Zugriffsschlüsseln zum Signieren von AWS CLI-, AWS API- oder AWS SDK-Anfragen gestellt wurden. Weitere Informationen zum Hinzufügen von MFA-Schutz zu API-Vorgängen mit temporären Sicherheitsanmeldedaten finden Sie unterKonfigurieren eines MFA-geschützten API-Zugriffs.
> Um zu überprüfen, ob MFA zur Validierung von IAM-Verbundidentitäten verwendet wird, können Sie die Authentifizierungsmethode von Ihrem Identitätsanbieter AWS als Sitzungs-Tag an übergeben. Details hierzu finden Sie unter Sitzungs-Tags übergeben AWS STS. Um MFA für IAM Identity Center-Identitäten durchzusetzen, können Sie Attribute für die Zugriffskontrolle aktivieren, um einen SAML-Assertion-Anspruch mit der Authentifizierungsmethode von Ihrem Identitätsanbieter an IAM Identity Center weiterzuleiten.

- Availability – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Auftraggeber temporäre Anmeldeinformationen für die Anforderung verwendet. Policen mit MFA-Bedingungen können beigefügt werden an:
  - Einem IAM-Benutzer oder einer IAM-Gruppe
  - Einer Ressource, wie zum Beispiel ein Amazon S3-Bucket, eine Amazon SQS-Warteschlange oder ein Amazon SNS-Thema
  - Die Vertrauensrichtlinie einer IAM-Rolle, die von einem Benutzer übernommen werden kann
- Datentyp — Boolean
- Werttyp - Einzelwertig

Temporäre Anmeldeinformationen werden verwendet, um IAM-Rollen und IAM-Benutzer mit temporären Tokens von AssumeRoleoder GetSessionToken sowie Benutzer von zu authentifizieren. AWS Management Console

Bei IAM-Benutzerzugriffsschlüsseln handelt es sich um langfristige Anmeldeinformationen. In einigen Fällen werden jedoch temporäre Anmeldeinformationen im Namen von IAM-Benutzern AWS erstellt, um Vorgänge auszuführen. In diesen Fällen ist der Schlüssel `aws:MultiFactorAuthPresent` in der Anforderung vorhanden und auf einen Wert von `false` gesetzt werden. Es gibt zwei häufige Fälle, in denen dies passieren kann:

- IAM-Benutzer in der Region verwenden AWS Management Console unwissentlich temporäre Anmeldeinformationen. Benutzer melden sich mit ihrem Benutzernamen und Passwort bei der Konsole an. Dabei handelt es sich um langfristige Anmeldeinformationen. Im Hintergrund erstellt die Konsole jedoch temporäre Anmeldeinformationen für den Benutzer.
- Wenn ein IAM-Benutzer einen Dienst aufruft, verwendet der AWS Dienst die Anmeldeinformationen des Benutzers erneut, um eine weitere Anfrage an einen anderen Dienst zu stellen. Zum Beispiel, wenn Sie Athena aufrufen, um auf einen Amazon S3 S3-Bucket zuzugreifen, oder wenn Sie es verwenden, AWS CloudFormation um eine Amazon EC2 EC2-Instance zu erstellen. AWS Verwendet für die nachfolgende Anfrage temporäre Anmeldeinformationen.

Informationen darüber, welche Services die Verwendung von temporären Anmeldeinformationen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren.

Der `aws:MultiFactorAuthPresent`-Schlüssel ist nie vorhanden, wenn ein API- oder CLI-Befehl mit langfristigen Anmeldeinformationen wie Zugriffsschlüsselpaare eines Benutzers aufgerufen wird. Wir empfehlen daher, beim Prüfen dieses Schlüssels die Versionen `...IfExists` der Bedingungsoperatoren zu verwenden.

Beachten Sie, dass das folgende `Condition`-Element keine zuverlässige Methode ist, um zu überprüfen, ob eine Anforderung über MFA authentifiziert wird.

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Diese Kombination aus `Deny` Effekt `Bool`-Element und `false`-Wert verweigert Anforderungen, die mit MFA authentifiziert werden können, es aber nicht waren. Dies gilt nur für temporäre Anmeldeinformationen, die die Verwendung von MFA unterstützen. Diese Anweisung verweigert nicht

den Zugriff auf Anforderungen, die mit langfristigen Anmeldeinformationen gestellt wurden, oder auf Anforderungen, die per MFA authentifiziert wurden. Verwenden Sie dieses Beispiel mit Vorsicht, da seine Logik kompliziert ist und nicht prüft, ob die MFA-Authentifizierung tatsächlich verwendet wurde.

Verwenden Sie auch nicht die Kombination aus `Deny` Effekt `Null`-Element und `true`, da sie sich genauso verhält und die Logik noch komplizierter ist.

Empfohlene Kombination

Wir empfehlen Ihnen, den Operator `BoolIfExists` zu verwenden, um zu überprüfen, ob eine Anforderung über MFA authentifiziert wird.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Diese Kombination aus `Deny`, `BoolIfExists` und `false` verweigert Anforderungen, die nicht mit MFA authentifiziert wurden. Insbesondere lehnt sie Anforderungen von temporären Anmeldeinformationen ab, die keine MFA umfassen. Es lehnt auch Anfragen ab, die mit langfristigen Anmeldeinformationen gestellt werden, wie AWS CLI z. B. AWS API-Operationen, die mit Zugriffsschlüsseln durchgeführt werden. Der Operator `*IfExists` prüft das Vorhandensein des `aws:MultiFactorAuthPresent`-Schlüssels und ob er vorhanden sein könnte oder nicht, wie durch seine Existenz angezeigt wird. Verwenden Sie diesen Operator, wenn Sie eine Anforderung ablehnen möchten, die nicht über die MFA authentifiziert wird. Dies ist sicherer, kann jedoch jeden Code oder jedes Skript beschädigen, das Zugriffsschlüssel für den Zugriff auf die AWS API AWS CLI oder verwendet.

Alternative Kombinationen

Sie können den `BoolIfExists`Operator auch verwenden, um MFA-authentifizierte Anfragen AWS CLI und/oder AWS API-Anfragen zuzulassen, die mit langfristigen Anmeldeinformationen gestellt werden.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Diese Bedingung ist erfüllt, wenn der Schlüssel existiert und vorhanden ist oder wenn der Schlüssel nicht vorhanden ist. Diese Kombination aus `Allow`, `BoolIfExists`, und `true` lässt Anforderungen zu, die mit MFA authentifiziert werden, oder Anforderungen, die nicht mit MFA authentifiziert werden können. Das bedeutet AWS CLI, dass AWS API- und AWS SDK-Operationen zulässig sind, wenn

der Anforderer seine langfristigen Zugriffsschlüssel verwendet. Diese Kombination erlaubt keine Anforderungen von temporären Anmeldeinformationen, die MFA umfassen könnten, aber keine MFA enthalten.

Wenn Sie eine Richtlinie mit dem visuellen Editor der IAM-Konsole erstellen und MFA required (MFA erforderlich), auswählen, wird diese Kombination angewendet. Diese Einstellung erfordert zwar MFA für den Konsolenzugriff, ermöglicht jedoch den programmgesteuerten Zugriff ohne MFA.

Alternativ können Sie den `Bool`-Operator verwenden, damit programmgesteuerte Anforderungen und Konsolenanforderungen nur dann zugelassen werden, wenn sie mit MFA authentifiziert werden.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Diese Kombination aus `Allow`, `Bool` und `true` lässt nur mit MFA authentifizierte Anforderungen zu. Dies gilt nur für temporäre Anmeldeinformationen, die die Verwendung von MFA unterstützen. Diese Anweisung erlaubt keinen Zugriff auf Anforderungen, die mit langfristigen Zugriffsschlüsseln gestellt wurden, oder auf Anforderungen, die mit temporären Anmeldeinformationen ohne MFA durchgeführt wurden.

Verwenden Sie keine Richtlinie wie die folgende Beispielrichtlinie, um auf das Vorhandensein des MFA-Schlüssels hin zu prüfen:

```
#####   WARNING: USE WITH CAUTION   #####

"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Diese Kombination aus `Allow` Effekt, `Null`-Element und `false`-Wert erlaubt nur Anforderungen, die mit MFA authentifiziert werden können, unabhängig davon, ob die Anforderung tatsächlich authentifiziert ist. Dies ermöglicht alle Anforderungen, die mit temporären Anmeldeinformationen gestellt werden, und verweigert den Zugriff mit langfristigen Anmeldeinformationen. Verwenden Sie dieses Beispiel mit Vorsicht, da es nicht prüft, ob die MFA-Authentifizierung tatsächlich verwendet wurde.

AWSInstanceSource: EC2 Vpc

Dieser Schlüssel identifiziert die VPC, an die Amazon-EC2-IAM-Rollen-Anmeldeinformationen übermittelt wurden. Sie können diesen Schlüssel in einer Richtlinie mit dem `aws:SourceVPC`globalen Schlüssel verwenden, um zu überprüfen, ob ein Anruf von einer VPC

(`aws:SourceVPC`) aus getätigt wird, die der VPC entspricht, an die die Anmeldeinformation übermittelt wurden (`aws:Ec2InstanceSourceVpc`).

- Verfügbarkeit – Dieser Schlüssel ist immer dann im Anforderungskontext enthalten, wenn der Anforderer Anforderungen mit einer Amazon EC2-Rollen-Anmeldeinformation signiert. Er kann in IAM-Richtlinien, Service-Control-Richtlinien, VPC-Endpunkt-Richtlinien und Ressourcenrichtlinien verwendet werden.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Dieser Schlüssel kann mit VPC-Identifikationswerten verwendet werden, ist jedoch am nützlichsten, wenn er als Variable in Kombination mit dem `aws:SourceVpc`-Kontextschlüssel verwendet wird. Dieser `aws:SourceVpc`-Kontextschlüssel ist nur dann im Anforderungskontext enthalten, wenn der Anforderer einen VPC-Endpunkt für die Anforderung verwendet. Die Verwendung von `aws:Ec2InstanceSourceVpc` mit `aws:SourceVpc` ermöglicht eine `aws:Ec2InstanceSourceVpc` breitere Verwendung, da Werte verglichen werden, die sich normalerweise zusammen ändern.

> **Note**
>
> Dieser Bedingungsschlüssel ist in EC2-Classic nicht verfügbar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
```

```
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Im obigen Beispiel wird der Zugriff verweigert, wenn der `aws:SourceVpc`-Wert nicht dem `aws:Ec2InstanceSourceVpc`-Wert entspricht. Die Richtlinienerklärung ist nur auf Rollen beschränkt, die als Amazon EC2-Instance-Rollen verwendet werden, indem getestet wird, ob der `ec2:SourceInstanceARN`-Bedingungsschlüssel vorhanden ist.

Die Richtlinie ermöglicht `aws:ViaAWSService` die Autorisierung von Anfragen AWS , wenn Anfragen im Namen Ihrer Amazon EC2 EC2-Instance-Rollen gestellt werden. Wenn Sie beispielsweise eine Anfrage von einer Amazon EC2 EC2-Instance an einen verschlüsselten Amazon S3-Bucket stellen, ruft Amazon S3 in Ihrem Namen AWS KMS an. Einige der Schlüssel sind nicht vorhanden, wenn die Anfrage an AWS KMS gestellt wird.

AWS: EC2 PrivateIPv4 InstanceSource

Dieser Schlüssel identifiziert die private IPv4-Adresse der primären elastischen Netzwerk-Schnittstelle, an die die Amazon EC2-IAM-Rollen-Anmeldeinformation übermittelt wurde. Sie müssen diesen Bedingungsschlüssel zusammen mit dem zugehörigen Schlüssel `aws:Ec2InstanceSourceVpc` verwenden, um sicherzustellen, dass Sie über eine global eindeutige Kombination aus VPC-ID und privater Quell-IP verfügen. Verwenden Sie diesen Schlüssel mit `aws:Ec2InstanceSourceVpc`, um sicherzustellen, dass eine Anfrage von derselben privaten IP-Adresse aus gestellt wurde, an die die Anmeldeinformation übermittelt wurde.

- Verfügbarkeit – Dieser Schlüssel ist immer dann im Anforderungskontext enthalten, wenn der Anforderer Anforderungen mit einer Amazon EC2-Rollen-Anmeldeinformation signiert. Er kann in IAM-Richtlinien, Service-Control-Richtlinien, VPC-Endpunkt-Richtlinien und Ressourcenrichtlinien verwendet werden.
- Datentyp — IP-Adresse
- Werttyp - Einzelwertig

Important

Dieser Schlüssel sollte nicht alleine in einer `Allow`-Anweisung verwendet werden. Private IP-Adressen sind per Definition nicht global eindeutig. Sie sollten den `aws:Ec2InstanceSourceVpc`-Schlüssel jedes Mal verwenden, wenn Sie den `aws:Ec2InstanceSourcePrivateIPv4`-Schlüssel verwenden, um die VPC anzugeben, von der aus Ihre Amazon EC2-Instance-Anmeldeinformation verwendet werden kann.

Note

Dieser Bedingungsschlüssel ist in EC2-Classic nicht verfügbar.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
```

```
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

als: SourceIdentity

Verwenden Sie diesen Schlüssel, um die Queltidentität, die vom Auftraggeber festgelegt wurde, mit der Quellidentität zu vergleichen, die Sie in der Richtlinie angeben.

- Verfügbarkeit — Dieser Schlüssel wird in den Anforderungskontext aufgenommen, nachdem eine Quellidentität festgelegt wurde, wenn eine Rolle mithilfe eines AWS STS Assume-Role-CLI-Befehls oder einer AWS STS `AssumeRole` API-Operation übernommen wird.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Sie können diesen Schlüssel in einer Richtlinie verwenden, um Aktionen AWS von Prinzipalen zuzulassen, die bei der Übernahme einer Rolle eine Quellidentität festgelegt haben. Die Aktivität für die angegebene Quellidentität der Rolle erscheint in AWS CloudTrail. Auf diese Weise können Administratoren leichter feststellen, wer oder was Aktionen mit einer Rolle in AWS ausgeführt hat.

Im Gegensatz zu `sts:RoleSessionName`, nachdem die Quellidentität festgelegt wurde, kann der Wert nicht geändert werden. Es ist im Kontext der Anforderung aller Aktionen vorhanden, die von der Rolle ausgeführt werden. Wenn Sie die Sitzungsanmeldeinformationen verwenden, bleibt der Wert in nachfolgenden Rollensitzungen bestehen, um eine andere Rolle anzunehmen. Die Annahme einer Rolle von einer anderen wird als Rollenverkettung bezeichnet.

Der `sts:SourceIdentity`Schlüssel ist in der Anfrage enthalten, wenn der Principal zunächst eine Quellidentität festlegt und gleichzeitig eine Rolle mithilfe eines AWS STS Assume-Role-CLI-Befehls oder einer AWS STS `AssumeRole` API-Operation annimmt. Der Schlüssel `aws:SourceIdentity` ist in der Anforderung für alle Aktionen vorhanden, die mit einer Rollensitzung durchgeführt werden, für die eine Quellidentität festgelegt wurde.

Die folgende Rollen-Vertrauensrichtlinie für `CriticalRole` im Konto `111122223333` enthält eine Bedingung für `aws:SourceIdentity`, die verhindert, dass ein Auftraggeber ohne eine Quellidentität, die auf Saanvi oder Diego eingestellt ist, die Rolle übernehmen kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Weitere Informationen zur Verwendung von Quellidentitätsinformationen finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

ec2: RoleDelivery

Verwenden Sie diesen Schlüssel, um die Version des Instance-Metadaten-Service in der signierten Anfrage mit den IAM-Rollenanmeldedaten für Amazon EC2 zu vergleichen. Der Instance-Metadaten-Service unterscheidet zwischen IMDSv1- und IMDSv2-Anforderungen, je nachdem, ob bei einer bestimmten Anforderung der `PUT`- oder `GET`-Header (die IMDSv2-spezifisch sind) vorhanden ist.

- Verfügbarkeit — Dieser Schlüssel ist immer dann im Anforderungskontext enthalten, wenn die Rollensitzung von einer Amazon EC2 EC2-Instance erstellt wird.
- Datentyp — Numerisch
- Werttyp - Einzelwertig
- Beispielwerte — 1.0, 2.0

Sie können den Instance Metadata Service (IMDS) auf jeder Instance so konfigurieren, dass lokaler Code oder Benutzer IMDSv2 verwenden müssen. Wenn Sie angeben, dass IMDSv2 verwendet werden muss, funktioniert IMDSv1 nicht mehr.

- Instance Metadata Service Version 1 (IMDSv1) — Eine Anforderungs-/Antwortmethode
- Instance-Metadatenservice Version 2 (IMDSv2) – Ein sitzungsorientiertes Verfahren

Informationen zur Konfiguration Ihrer Instance für die Verwendung von IMDSv2 finden Sie unter Konfiguration der Instanz-Metadatenoptionen.

Im folgenden Beispiel wird der Zugriff verweigert, wenn der RoleDelivery Wert ec2: im Anforderungskontext 1,0 (IMDSv1) ist. Diese Richtlinienerklärung kann allgemein angewendet werden, da die Anforderung keine Wirkung hat, wenn sie nicht mit Amazon EC2 EC2-Rollenanmeldedaten signiert ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Weitere Informationen finden Sie unter Beispielrichtlinien für die Arbeit mit Instance-Metadaten.

ec2: Arn SourceInstance

Verwenden Sie diesen Schlüssel, um den ARN der Instanz zu vergleichen, von der aus die Sitzung der Rolle generiert wurde.

- Verfügbarkeit — Dieser Schlüssel ist immer dann im Anforderungskontext enthalten, wenn die Rollensitzung von einer Amazon EC2 EC2-Instance erstellt wird.

- Datentyp — ARN
- Werttyp - Einzelwertig
- Beispielwert — arn:aws:ec2:us-west- 2:111111111111:instance/instance-id

Richtlinienbeispiele finden Sie unter Zulassen, dass eine bestimmte Instanz Ressourcen in anderen Diensten anzeigt. AWS

Klebstoff: RoleAssumed Von

Der AWS Glue Service legt diesen Bedingungsschlüssel für jede AWS API-Anfrage fest, bei der eine Anfrage mithilfe einer Servicerolle im Namen des Kunden gestellt wird (nicht durch einen Job- oder Entwickler-Endpunkt, sondern direkt durch den AWS Glue Service). AWS Glue Verwenden Sie diesen Schlüssel, um zu überprüfen, ob ein Anruf an eine AWS Ressource vom AWS Glue Dienst kam.

- Verfügbarkeit — Dieser Schlüssel ist im Anforderungskontext enthalten, AWS Glue wenn eine Anfrage mithilfe einer Servicerolle im Namen des Kunden gestellt wird.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig
- Beispielwert — Dieser Schlüssel ist immer auf gesetzt`glue.amazonaws.com`.

Das folgende Beispiel fügt eine Bedingung hinzu, die es dem AWS Glue Service ermöglicht, ein Objekt aus einem Amazon S3 S3-Bucket abzurufen.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

Klebstoff: CredentialIssuing Service

Der AWS Glue Dienst legt diesen Schlüssel für jede AWS API-Anfrage fest und verwendet dabei eine Servicerolle, die von einem Job- oder Entwickler-Endpunkt stammt. Verwenden Sie diesen Schlüssel, um zu überprüfen, ob ein Aufruf einer AWS Ressource von einem AWS Glue Job- oder Entwickler-Endpunkt aus erfolgte.

- Verfügbarkeit — Dieser Schlüssel ist im Anforderungskontext enthalten, wenn eine AWS Glue Anfrage gestellt wird, die von einem Job- oder Entwickler-Endpunkt kommt.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig
- Beispielwert — Dieser Schlüssel ist immer auf gesetzt`glue.amazonaws.com`.

Im folgenden Beispiel wird eine Bedingung hinzugefügt, die an eine IAM-Rolle angehängt ist, die von einem AWS Glue Job verwendet wird. Dadurch wird sichergestellt, dass bestimmte Aktionen je nachdem, ob die Rollensitzung für eine AWS Glue Job-Laufzeitumgebung verwendet wird, erlaubt/verweigert werden.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

Lambda: Arn SourceFunction

Verwenden Sie diesen Schlüssel, um den ARN der Lambda-Funktion zu identifizieren, an den die Anmeldeinformationen für die IAM-Rolle übermittelt wurden. Der Lambda-Dienst legt diesen Schlüssel für jede AWS API-Anfrage fest, die aus der Ausführungsumgebung Ihrer Funktion stammt. Verwenden Sie diesen Schlüssel, um zu überprüfen, ob ein Aufruf einer AWS Ressource aus dem Code einer bestimmten Lambda-Funktion stammt. Lambda legt diesen Schlüssel auch für einige Anfragen fest, die von außerhalb der Ausführungsumgebung kommen, z. B. das Schreiben von Protokollen in X-Ray CloudWatch und das Senden von Traces an X-Ray.

- Verfügbarkeit — Dieser Schlüssel ist im Anforderungskontext enthalten, wenn der Lambda-Funktionscode aufgerufen wird.
- Datentyp — ARN
- Werttyp - Einzelwertig
- Beispielwert — arn:aws:lambda:us-east- 1:123456789012:function: TestFunction

Das folgende Beispiel ermöglicht einer bestimmten Lambda-Funktion den `s3:PutObject` Zugriff auf den angegebenen Bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

Weitere Informationen finden Sie unter Working with Lambda Execution Environment Credentials im AWS Lambda Developer Guide.

ssm: Arn SourceInstance

Verwenden Sie diesen Schlüssel, um den ARN der AWS Systems Manager verwalteten Instanz zu identifizieren, an den die Anmeldeinformationen für die IAM-Rolle übermittelt wurden. Dieser Bedingungsschlüssel ist nicht vorhanden, wenn die Anfrage von einer verwalteten Instance mit einer IAM-Rolle stammt, die einem Amazon EC2 EC2-Instance-Profil zugeordnet ist.

- Verfügbarkeit — Dieser Schlüssel ist immer dann im Anforderungskontext enthalten, wenn Rollenanmeldedaten an eine AWS Systems Manager verwaltete Instance übermittelt werden.
- Datentyp — ARN

- Werttyp - Einzelwertig
- Beispielwert — arn:aws:ec2:us-west- 2:111111111111:instance/instance-id

Identitätsspeicher: UserId

Verwenden Sie diesen Schlüssel, um die Identität der IAM Identity Center-Mitarbeiter in der signierten Anfrage mit der in der Richtlinie angegebenen Identität zu vergleichen.

- Verfügbarkeit — Dieser Schlüssel ist enthalten, wenn der Anrufer der Anfrage ein Benutzer in IAM Identity Center ist.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig
- Beispielwert — 94482488-3041-7026-18f3-be45837cd0e4

Sie können den Namen eines Benutzers im IAM Identity Center ermitteln, indem Sie mit UserId der API oder dem SDK eine Anfrage an die ID-API stellen. GetUser AWS CLI AWS AWS

Eigenschaften des Netzwerks

Verwenden Sie die folgenden Bedingungsschlüssel, um Details über das Netzwerk, von dem die Anfrage stammt oder über das die Anfrage weitergeleitet wurde, mit den Netzwerkeigenschaften zu vergleichen, die Sie in der Richtlinie angeben.

Inhalt

- aws: SourceIp
- aws: SourceVpc
- als: SourceVpce
- aws: VpcSource Ip

aws: SourceIp

Verwenden Sie diesen Schlüssel, um die IP-Adresse des Anforderers mit der IP-Adresse zu vergleichen, die Sie in der Richtlinie angeben. Der `aws:SourceIp`-Bedingungsschlüssel kann nur für öffentliche IP-Adressbereiche verwendet werden.

- Verfügbarkeit – Dieser Schlüssel ist im Anforderungskontext enthalten, es sei denn, der Anforderer verwendet einen VPC-Endpunkt für die Anforderung.

- Datentyp — IP-Adresse
- Werttyp - Einzelwertig

Der `aws:SourceIp`-Bedingungsschlüssel kann in einer Richtlinie verwendet werden, um Auftraggeber zu erlauben, Anforderungen nur innerhalb eines angegebenen IP-Bereichs zu stellen.

> **Note**
>
> `aws:SourceIp` unterstützt IPv4- und IPv6-IP-Adressen oder einen Bereich von IP-Adressen. Eine Liste derjenigen AWS-Services , die IPv6 unterstützen AWS-Services , finden Sie im Amazon VPC-Benutzerhandbuch unter IPv6-Unterstützung.

Sie können beispielsweise die folgende identitätsbasierte Richtlinie an eine IAM-Rolle anfügen. Mit dieser Richtlinie kann der Benutzer Objekte in den Amazon-S3-Bucket `DOC-EXAMPLE-BUCKET3` einfügen, wenn er den Aufruf über den angegebenen IPv4-Adressbereich tätigt. Diese Richtlinie erlaubt auch einem AWS Service, der diesen Vorgang Forward Access Sessions (FAS) in Ihrem Namen durchführt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        }
    ]
}
```

Wenn Sie den Zugriff von Netzwerken einschränken müssen, die sowohl IPv4- als auch IPv6-Adressierung unterstützen, können Sie die IPv4- und IPv6-Adresse oder IP-Adressbereiche in die IAM-Richtlinienbedingung einschließen. Mit der folgenden identitätsbasierten Richtlinie kann

der Benutzer Objekte in den Amazon-S3-Bucket `DOC-EXAMPLE-BUCKET3` einfügen, wenn er den Aufruf von einem der angegebenen IPv4- oder IPv6-Adressbereiche aus tätigt. Bevor Sie IPv6-Adressbereiche in Ihre IAM-Richtlinie aufnehmen, stellen Sie sicher, dass der, mit dem AWS-Service Sie arbeiten, IPv6 unterstützt. Eine Liste derjenigen AWS-Services , die IPv6 unterstützen AWS-Services , finden Sie im Amazon VPC-Benutzerhandbuch unter IPv6-Unterstützung.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Wenn die Anforderung von einem Host stammt, der einen Amazon VPC-Endpunkt verwendet, ist der Schlüssel `aws:SourceIp` nicht verfügbar. Sie sollten stattdessen einen VPC-spezifischen Schlüssel wie aws: VpcSource Ip verwenden. Weitere Informationen zur Verwendung von VPC-Endpunkten finden Sie unter Identitäts- und Zugriffsmanagement für VPC-Endpunkte und VPC-Endpunkt-Services im AWS PrivateLink -Leitfaden.

aws: SourceVpc

Verwenden Sie diesen Schlüssel, um zu überprüfen, ob die Anfrage die VPC durchläuft, an die der VPC-Endpunkt angeschlossen ist. In einer Richtlinie können Sie diesen Schlüssel verwenden, um den Zugriff auf nur eine bestimmte VPC zu erlauben. Weitere Informationen finden Sie unter Beschränkung des Zugriffs auf eine bestimmte VPC im Benutzerhandbuch für Amazon Simple Storage Service.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Anforderer einen VPC-Endpunkt für die Anforderung verwendet.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

als: SourceVpce

Verwenden Sie diesen Schlüssel, um die VPC-Endpunkt-ID der Anforderung mit der Endpunkt-ID zu vergleichen, die Sie in der Richtlinie angeben. In einer Richtlinie können Sie diesen Schlüssel verwenden, um den Zugriff auf einen bestimmten VPC-Endpunkt zu beschränken. Weitere Informationen finden Sie unter Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt im Benutzerhandbuch für Amazon Simple Storage Service.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Anforderer einen VPC-Endpunkt für die Anforderung verwendet.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

aws: VpcSource Ip

Verwenden Sie diesen Schlüssel, um die IP-Adresse, von der eine Anforderung stammt, mit der IP-Adresse zu vergleichen, die Sie in der Richtlinie angeben. In einer Richtlinie stimmt der Schlüssel nur dann überein, wenn die Anforderung von der angegebenen IP-Adresse stammt und über einen VPC-Endpunkt geleitet wird.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn die Anforderung über einen VPC-Endpunkt erfolgt.
- Datentyp — IP-Adresse
- Werttyp - Einzelwertig

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon VPC User Guide.

> **Note**
>
> `aws:VpcSourceIp` unterstützt IPv4- und IPv6-IP-Adressen oder einen Bereich von IP-Adressen. Eine Liste derjenigen AWS-Services , die IPv6 unterstützen AWS-Services , finden Sie im Amazon VPC-Benutzerhandbuch unter IPv6-Unterstützung.

Eigenschaften der Ressource

Verwenden Sie die folgenden Bedingungsschlüssel, um Details über die Ressource, die das Ziel der Anfrage ist, mit den Ressourceneigenschaften zu vergleichen, die Sie in der Richtlinie angeben.

Inhalt

aws: ResourceAccount

Verwenden Sie diesen Schlüssel, um die AWS-Konto -ID des angeforderten Ressourcenbesitzers mit dem Ressourcenkonto in der Richtlinie zu vergleichen. Sie können dann den Zugriff auf diese Ressource basierend auf dem Konto, dem die Ressource gehört, erlauben oder verweigern.

- Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext für die meisten Service-Aktionen enthalten. Die folgenden Aktionen unterstützen diesen Schlüssel nicht:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store – Alle Aktionen
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`

  - `ec2:CopyImage`
  - `ec2:CopySnapshot`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVolume`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
  - `ec2:DeleteTransitGatewayPeeringAttachment`
  - `ec2:DeleteVpcPeeringConnection`
  - `ec2:RejectTransitGatewayPeeringAttachment`
  - `ec2:RejectVpcEndpointConnections`
  - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` ruft einen Event-Bus in einem anderen Konto auf, wenn dieser Event-Bus vor dem 2. März 2023 als kontenübergreifendes EventBridge Ziel konfiguriert wurde. Weitere Informationen finden Sie im EventBridge Amazon-Benutzerhandbuch unter Erteilen von Berechtigungen zum Zulassen von Ereignissen aus anderen AWS Konten.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Amazon-Dienst
  - `es:AcceptInboundConnection`
  - `es:CreateOutboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`
  - `route53:ListHostedZonesByVPC`
- AWS Security Hub

    - `securityhub:AcceptAdministratorInvitation`
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

> **Note**
>
> Weitere Überlegungen zu den oben genannten nicht unterstützten Aktionen finden Sie im Repository Data Perimeter Policy Examples (Beispiele für Datenperimeter-Richtlinien).

Dieser Schlüssel entspricht der AWS-Konto ID für das Konto, dessen Ressourcen in der Anfrage ausgewertet wurden.

Für die meisten Ressourcen in Ihrem Konto, enthält der ARN die Besitzer-Konto-ID für diese Ressource. Für bestimmte Ressourcen, wie Amazon S3-Buckets, enthält der Ressourcen-ARN nicht die Konto-ID. Die folgenden zwei Beispiele zeigen den Unterschied zwischen einer Ressource mit einer Konto-ID im ARN und einem Amazon-S3-ARN ohne Konto-ID:

- `arn:aws:iam::123456789012:role/AWSExampleRole` – Erstellung und Eigentümerschaft der IAM-Rolle innerhalb des Kontos 123456789012.
- `arn:aws:s3:::DOC-EXAMPLE-BUCKET2` – Erstellung und Eigentümerchaft des Amazon-S3-Buckets innerhalb des Kontos `111122223333`, wird nicht im ARN angezeigt.

Verwenden Sie die AWS Konsole, API oder CLI, um alle Ihre Ressourcen und die entsprechenden ARNs zu finden.

Sie schreiben eine Richtlinie, die Berechtigungen für Ressourcen basierend auf der Konto-ID des Ressourcenbesitzers verweigert. Zum Beispiel verweigert die folgende identitätsbasierte Richtlinie den Zugriff auf die angegebene Ressource, wenn die Ressource nicht dem angegebenen Konto angehört.

Um diese Richtlinie zu verwenden, ersetzen Sie den kursiv gedruckten Platzhaltertext durch Ihre eigenen Kontoinformationen.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Stattdessen wird der `Deny`-Effekt verwendet, der explizit den Zugriff auf alle in der Anweisung aufgeführten Ressourcen verweigert, die nicht zu dem aufgelisteten Konto gehören. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die den Zugriff auf bestimmte Ressourcen gewähren.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:account:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "account"
          ]
        }
      }
    }
  ]
}
```

Diese Richtlinie verweigert den Zugriff auf alle Ressourcen für einen bestimmten AWS Dienst, es sei denn, der angegebene AWS-Konto Eigentümer der Ressource ist.

> **Note**
>
> Einige AWS-Services erfordern den Zugriff auf AWS eigene Ressourcen, die in einem anderen AWS-Konto gehostet werden. Wenn Sie `aws:ResourceAccount` in Ihren identitätsbasierten Richtlinien verwenden, kann sich dies auf die Fähigkeit Ihrer Identität auswirken, auf diese Ressourcen zuzugreifen.

Bestimmte AWS Dienste, wie z. B. AWS Data Exchange, sind AWS-Konten für den normalen Betrieb auf den Zugriff auf Ressourcen außerhalb Ihres Computers angewiesen. Wenn Sie das Element `aws:ResourceAccount` in Ihren Richtlinien verwenden, fügen Sie zusätzliche Erklärungen hinzu, um Ausnahmen für diese Services zu erstellen. Die Beispielrichtlinie AWS: Verweigern Sie den Zugriff auf Amazon S3 S3-Ressourcen außerhalb Ihres Kontos, außer AWS Data Exchange zeigt, wie der Zugriff basierend auf dem Ressourcenkonto verweigert wird, während Ausnahmen für serviceeigene Ressourcen definiert werden.

Verwenden Sie diese Richtlinienbeispiele als Vorlagen für die Erstellung Ihrer eigenen benutzerdefinierten Richtlinien. Weitere Informationen finden Sie in Ihrer Service-Dokumentation.

aws: ResourceOrg Pfade

Verwenden Sie diesen Schlüssel, um den AWS Organisationspfad für die Ressource, auf die zugegriffen wurde, mit dem Pfad in der Richtlinie zu vergleichen. In einer Richtlinie stellt dieser Bedingungsschlüssel sicher, dass die Ressource einem Kontomitglied innerhalb des angegebenen Organisationsstamms oder der Organisationseinheiten (OUs) in AWS Organizations gehört. Ein AWS Organisationspfad ist eine Textdarstellung der Struktur einer Organisationseinheit. Weitere Informationen zum Verwenden und Verstehen von Pfaden finden Sie unter Der AWS Organizations - Entitätspfad.

- Verfügbarkeit – Dieser Schlüssel wird nur dann in den Anforderungskontext aufgenommen, wenn das Konto, dem die Ressource gehört, Mitglied einer Organisation ist. Dieser globale Bedingungsschlüssel unterstützt die folgenden Aktionen nicht:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store – Alle Aktionen
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`

    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`
    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
    - `events:PutEvents`— EventBridge `PutEvents` ruft einen Event-Bus in einem anderen Konto auf, wenn dieser Event-Bus vor dem 2. März 2023 als kontenübergreifendes EventBridge Ziel konfiguriert wurde. Weitere Informationen finden Sie im EventBridge Amazon-Benutzerhandbuch unter Erteilen von Berechtigungen zum Zulassen von Ereignissen aus anderen AWS Konten.
- Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
    - `macie2:AcceptInvitation`
- OpenSearch Amazon-Dienst
    - `es:AcceptInboundConnection`
    - `es:CreateOutboundConnection`
- Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
- AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`

- Datentyp — Zeichenfolge (Liste)
- Werttyp - Mehrwertig

> **Note**
>
> Weitere Überlegungen zu den oben genannten nicht unterstützten Aktionen finden Sie im Repository Data Perimeter Policy Examples (Beispiele für Datenperimeter-Richtlinien).

`aws:ResourceOrgPaths` ist ein mehrwertiger Bedingungsschlüssel. Mehrwertige Bedingungsschlüssel können im Anforderungskontext mehrere Werte haben. Sie müssen die Set-Operatoren `ForAnyValue` oder `ForAllValues` zusammen mit String-Bedingungsoperatoren für diesen Schlüssel verwenden. Weitere Hinweise zu mehrwertigen Bedingungsschlüsseln finden Sie unter Mehrwertige Kontextschlüssel.

Zum Beispiel gibt die folgende Bedingung `True` für Ressourcen zurück, die zur Organisation `o-a1b2c3d4e5` gehören. Wenn Sie einen Platzhalter angeben, müssen Sie den StringLikeBedingungsoperator verwenden.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

Die folgende Bedingung gibt `True` für Ressourcen mit der OU-ID `ou-ab12-11111111` zurück. Es werden Ressourcen abgestimmt, die Konten gehören, die der OU ou-ab12-11111111 oder einer der untergeordneten OUs zugeordnet sind.

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

Die folgende Bedingung gibt `True` für Ressourcen zurück, die sich im Besitz von Konten befinden, die direkt der OU-ID `ou-ab12-22222222` angefügt sind, aber nicht ihren untergeordneten OUs. Im folgenden Beispiel wird der StringEqualsBedingungsoperator verwendet, um die exakte Übereinstimmungsanforderung für die OU-ID und nicht die Übereinstimmung mit Platzhaltern anzugeben.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> Einige AWS-Services benötigen Zugriff auf AWS eigene Ressourcen, die auf anderen AWS-Konto gehostet werden. Wenn Sie `aws:ResourceOrgPaths` in Ihren identitätsbasierten Richtlinien verwenden, kann sich dies auf die Fähigkeit Ihrer Identität auswirken, auf diese Ressourcen zuzugreifen.

Bestimmte AWS Dienste, wie z. B. AWS Data Exchange, sind AWS-Konten für den normalen Betrieb auf den Zugriff auf Ressourcen außerhalb Ihres Computers angewiesen. Wenn Sie den Schlüssel `aws:ResourceOrgPaths` in Ihren Richtlinien verwenden, fügen Sie zusätzliche Erklärungen hinzu, um Ausnahmen für diese Services zu erstellen. Die Beispielrichtlinie AWS: Verweigern Sie den Zugriff auf Amazon S3 S3-Ressourcen außerhalb Ihres Kontos, außer AWS Data Exchange zeigt, wie der Zugriff basierend auf dem Ressourcenkonto verweigert wird, während Ausnahmen für serviceeigene Ressourcen definiert werden. Sie können eine ähnliche Richtlinie erstellen, um den Zugriff auf Ressourcen innerhalb Ihrer Organisationseinheit (OU) mithilfe des `aws:ResourceOrgPaths`-Schlüssels zu beschränken und gleichzeitig serviceeigene Ressourcen zu berücksichtigen.

Verwenden Sie diese Richtlinienbeispiele als Vorlagen für die Erstellung Ihrer eigenen benutzerdefinierten Richtlinien. Weitere Informationen finden Sie in Ihrer Service-Dokumentation.

aws: ResourceOrg ID

Verwenden Sie diesen Schlüssel, um die ID der Organisation in AWS Organizations, zu der die angeforderte Ressource gehört, mit der in der Richtlinie angegebenen ID zu vergleichen.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn das Konto, das Eigentümer der Ressource ist, Mitglied einer Organisation ist. Dieser globale Bedingungsschlüssel unterstützt die folgenden Aktionen nicht:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store – Alle Aktionen
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`

- `ec2:AcceptVpcEndpointConnections`
- `ec2:AcceptVpcPeeringConnection`
- `ec2:CopyImage`
- `ec2:CopySnapshot`
- `ec2:CreateTransitGatewayPeeringAttachment`
- `ec2:CreateVolume`
- `ec2:CreateVpcEndpoint`
- `ec2:CreateVpcPeeringConnection`
- `ec2:DeleteTransitGatewayPeeringAttachment`
- `ec2:DeleteVpcPeeringConnection`
- `ec2:RejectTransitGatewayPeeringAttachment`
- `ec2:RejectVpcEndpointConnections`
- `ec2:RejectVpcPeeringConnection`

- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` ruft einen Event-Bus in einem anderen Konto auf, wenn dieser Event-Bus vor dem 2. März 2023 als kontenübergreifendes EventBridge Ziel konfiguriert wurde. Weitere Informationen finden Sie im EventBridge Amazon-Benutzerhandbuch unter Erteilen von Berechtigungen zum Zulassen von Ereignissen aus anderen AWS Konten.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Amazon-Dienst
  - `es:AcceptInboundConnection`
  - `es:CreateOutboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`

    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

> **Note**
>
> Weitere Überlegungen zu den oben genannten nicht unterstützten Aktionen finden Sie im Repository Data Perimeter Policy Examples (Beispiele für Datenperimeter-Richtlinien).

Dieser globale Schlüssel gibt die ID der Ressourcenorganisation für eine bestimmte Anforderung zurück. Es erlaubt Ihnen, Regeln zu erstellen, die für alle Ressourcen in einer Organisation gelten, die im `Resource`-Element einer identitätsbasierten Richtlinie angegeben sind. Sie können die Organisations-ID im Bedingungselement angeben. Wenn Sie Konten hinzufügen und entfernen, schließen Richtlinien, die den `aws:ResourceOrgID`-Schlüssel enthalten, automatisch die richtigen Konten ein und Sie müssen sie nicht manuell aktualisieren.

Zum Beispiel verhindert die folgende Richtlinie, dass der Prinzipal der `policy-genius-dev`-Ressource Objekte hinzufügt, es sei denn, die Amazon-S3-Ressource gehört derselben Organisation an wie der Prinzipal, der die Anforderung stellt.

> **Important**
>
> Diese Richtlinie lässt keine Aktionen zu. Stattdessen wird der `Deny`-Effekt verwendet, der explizit den Zugriff auf alle in der Anweisung aufgeführten Ressourcen verweigert, die nicht zu dem aufgelisteten Konto gehören. Verwenden Sie diese Richtlinie in Kombination mit anderen Richtlinien, die den Zugriff auf bestimmte Ressourcen gewähren.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
    "Effect": "Deny",
    "Action": "s3:PutObject",
```

```
    "Resource": "arn:partition:s3:::policy-genius-dev/*",
    "Condition": {
      "StringNotEquals": {
        "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
      }
    }
  }
}
```

> **Note**
>
> Einige AWS-Services benötigen Zugriff auf AWS eigene Ressourcen, die auf anderen gehostet werden AWS-Konto. Wenn Sie `aws:ResourceOrgID` in Ihren identitätsbasierten Richtlinien verwenden, kann sich dies auf die Fähigkeit Ihrer Identität auswirken, auf diese Ressourcen zuzugreifen.

Bestimmte AWS Dienste, wie z. B. AWS Data Exchange, sind AWS-Konten für den normalen Betrieb auf den Zugriff auf Ressourcen außerhalb Ihres Computers angewiesen. Wenn Sie den Schlüssel `aws:ResourceOrgID` in Ihren Richtlinien verwenden, fügen Sie zusätzliche Erklärungen hinzu, um Ausnahmen für diese Services zu erstellen. Die Beispielrichtlinie AWS: Verweigern Sie den Zugriff auf Amazon S3 S3-Ressourcen außerhalb Ihres Kontos, außer AWS Data Exchange zeigt, wie der Zugriff basierend auf dem Ressourcenkonto verweigert wird, während Ausnahmen für serviceeigene Ressourcen definiert werden. Sie können eine ähnliche Richtlinie erstellen, um den Zugriff auf Ressourcen innerhalb Ihrer Organisation mithilfe des `aws:ResourceOrgID`-Schlüssels zu beschränken und gleichzeitig serviceeigene Ressourcen zu berücksichtigen.

Verwenden Sie diese Richtlinienbeispiele als Vorlagen für die Erstellung Ihrer eigenen benutzerdefinierten Richtlinien. Weitere Informationen finden Sie in Ihrer Service-Dokumentation.

Im folgenden Video erfahren Sie mehr darüber, wie Sie den Bedingungsschlüssel `aws:ResourceOrgID` in einer Richtlinie verwenden können.

Sicherstellen, dass Identitäten und Netzwerke nur für den Zugriff auf vertrauenswürdige Ressourcen verwendet werden können.

aws:ResourceTag/Tag-Taste

Verwenden Sie diesen Schlüssel, um das Tag-Schlüssel-Wert-Paar, das Sie in der Richtlinie angeben, mit dem Schlüssel-Wert-Paar zu vergleichen, das der Ressource zugeordnet ist. Beispiel:

Sie können verlangen, dass der Zugriff auf eine Ressource nur gewährt wird, wenn die Ressource über den angefügten Tag-Schlüssel `"Dept"` mit dem Wert `"Marketing"` verfügt. Weitere Informationen finden Sie unter Steuern des Zugriffs auf AWS -Ressourcen.

- Verfügbarkeit – Dieser Schlüssel wird in den Anforderungskontext aufgenommen, wenn die angeforderte Ressource bereits angefügte Tags hat oder in Anforderungen, die eine Ressource mit einem angefügten Tag erstellen. Dieser Schlüssel wird nur für Ressourcen zurückgegeben, die Berechtigung auf Basis von Tags unterstützen. Es gibt einen Kontextschlüssel für jedes Tag-Schlüssel-Wert-Paar.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

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. Bei Tag-Schlüsseln und -Werten wird nicht zwischen Groß- und Kleinschreibung unterschieden. Dies bedeutet Folgendes: Wenn Sie `"aws:ResourceTag/TagKey1": "Value1"` im Bedingungselement Ihrer Richtlinie angeben, stimmt die Bedingung mit einem Ressourcen-Tag-Schlüssel mit dem Namen `TagKey1` oder `tagkey1` überein, aber nicht mit beiden.

Beispiele für die Verwendung des `aws:ResourceTag`-Schlüssels zur Steuerung des Zugriffs auf IAM-Ressourcen finden Sie unter Steuern des Zugriffs auf AWS -Ressourcen.

Beispiele für die Verwendung des `aws:ResourceTag` Schlüssels zur Steuerung des Zugriffs auf andere AWS Ressourcen finden Sie unterSteuern des Zugriffs auf AWS Ressourcen mithilfe von Tags.

Ein Tutorial zur Verwendung des `aws:ResourceTag`-Bedingungsschlüssels für attributbasierte Zugriffskontrolle (ABAC) finden Sie unter IAM-Tutorial: Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags definieren.

Eigenschaften der Anfrage

Verwenden Sie die folgenden Bedingungsschlüssel, um Details zur Anfrage selbst und zum Inhalt der Anforderung mit den Anforderungseigenschaften zu vergleichen, die Sie in der Richtlinie angeben.

Inhalt

- aws: CalledVia
- aws: CalledVia Zuerst

- aws: CalledVia Zuletzt
- AWS: via AWSService
- war: CurrentTime
- als: EpochTime
- aws:referer
- als: RequestedRegion
- RequestTagaws/Tag-Schlüssel
- als: TagKeys
- aws: SecureTransport
- war: SourceArn
- aws: SourceAccount
- aws: SourceOrg Pfade
- aws: SourceOrg ID
- aws: UserAgent

aws: CalledVia

Verwenden Sie diesen Schlüssel, um die Services in der Richtlinie mit den Services zu vergleichen, die im Auftrag des IAM-Auftraggebers (Benutzer oder Rolle) Anforderungen ausgegeben haben. Wenn ein Principal eine Anfrage an einen AWS Dienst stellt, verwendet dieser Dienst möglicherweise die Anmeldeinformationen des Prinzipals, um nachfolgende Anfragen an andere Dienste zu stellen. Der Schlüssel `aws:CalledVia` enthält eine geordnete Liste aller Services in der Kette, die im Auftrag des Auftraggebers Anforderungen ausgegeben haben.

Sie können es beispielsweise verwenden, AWS CloudFormation um aus einer Amazon DynamoDB-Tabelle zu lesen und zu schreiben. DynamoDB verwendet dann die von AWS Key Management Service ()AWS KMS bereitgestellte Verschlüsselung.

- Verfügbarkeit – Dieser Schlüssel ist in der Anforderung vorhanden, wenn ein Service, der `aws:CalledVia` unterstützt, die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um eine Anforderung an einen anderen Service auszugeben. Dieser Schlüssel ist nicht vorhanden, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen. Dieser Schlüssel ist auch nicht vorhanden, wenn der Auftraggeber den Aufruf direkt durchführt.

- Datentyp — Zeichenfolge (Liste)
- Werttyp - Mehrwertig

Um den `aws:CalledVia` Bedingungsschlüssel in einer Richtlinie verwenden zu können, müssen Sie die Dienstprinzipale angeben, um AWS Serviceanfragen zuzulassen oder abzulehnen. AWS unterstützt die Verwendung der folgenden Dienstprinzipale mit. `aws:CalledVia`

Dienstauftraggeber
aoss.amazonaws.com
athena.amazonaws.com
backup.amazonaws.com
cloud9.amazonaws.com
cloudformation.amazonaws.com
databrew.amazonaws.com
dataexchange.amazonaws.com
dynamodb.amazonaws.com
imagebuilder.amazonaws.com
kms.amazonaws.com
mgn.amazonaws.com
nimble.amazonaws.com
omics.amazonaws.com
ram.amazonaws.com
robomaker.amazonaws.com
servicecatalog-appregistry.amazonaws.com

Dienstauftraggeber
sqlworkbench.amazonaws.com
ssm-guiconnect.amazonaws.com

Um den Zugriff zu erlauben oder zu verweigern, wenn irgendein Service unter Verwendung der Anmeldeinformationen des Auftraggebers eine Anforderung ausgibt, verwenden Sie den Bedingungsschlüssel `AWS: via AWSService`. Dieser Bedingungsschlüssel unterstützt AWS Dienste.

Der `aws:CalledVia`-Schlüssel ist ein mehrwertiger Schlüssel. Sie können die Reihenfolge jedoch nicht mit diesem Schlüssel in einer Bedingung erzwingen. Anhand des obigen Beispiels gibt User 1 (Benutzer 1) eine Anforderung an AWS CloudFormation aus, das DynamoDB aufruft, das wiederum AWS KMS aufruft. Dies sind drei separate Anforderungen. Der letzte Aufruf von AWS KMS wird von Benutzer 1 über AWS CloudFormation und dann DynamoDB ausgeführt.

In diesem Fall enthält der Schlüssel `aws:CalledVia` im Anforderungskontext `cloudformation.amazonaws.com` und `dynamodb.amazonaws.com` (in dieser Reihenfolge). Wenn Ihnen nur wichtig ist, dass der Aufruf über DynamoDB irgendwo in der Kette von Anforderungen erfolgt ist, können Sie diesen Bedingungsschlüssel in Ihrer Richtlinie verwenden.

Die folgende Richtlinie ermöglicht beispielsweise die Verwaltung des genannten AWS KMS Schlüsselsmy`-example-key`, jedoch nur, wenn DynamoDB einer der anfordernden Dienste ist. Der Bedingungsoperator `ForAnyValue:StringEquals` stellt sicher, dass DynamoDB einer der aufrufenden Services ist. Wenn der Auftraggeber AWS KMS direkt aufruft, gibt die Bedingung `false` zurück, und die Anforderung wird von dieser Richtlinie nicht zugelassen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
```

```
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": ["dynamodb.amazonaws.com"]
                }
            }
        }
    ]
}
```

Mit den Schlüsseln aws:CalledViaFirst und aws:CalledViaLast können Sie auf Wunsch erzwingen, welcher Service den ersten oder letzten Aufruf in der Kette durchführt. Die folgende Richtlinie ermöglicht beispielsweise die Verwaltung des in genannten `my-example-key` Schlüssels. AWS KMS Diese AWS KMS Operationen sind nur zulässig, wenn mehrere Anfragen in der Kette enthalten waren. Die erste Anforderung muss über AWS CloudFormation und die letzte über DynamoDB erfolgen. Wenn andere Services mitten in der Kette Anforderungen ausgeben, ist die Operation trotzdem zulässig.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
```

```
}
```

Die Schlüssel `aws:CalledViaFirst` und `aws:CalledViaLast` sind in der Anforderung vorhanden, wenn ein Service die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um einen anderen Service aufzurufen. Sie geben die ersten und letzten Services an, die in der Kette von Anforderungen Aufrufe durchgeführt haben. Nehmen wir beispielsweise an, dass AWS CloudFormation ein anderer Dienst mit dem Namen`X Service`, aufgerufen wird, der DynamoDB aufruft, der dann aufruft. AWS KMS Der letzte Aufruf von AWS KMS erfolgt durch `User 1` via AWS CloudFormation`X Service`, then und dann DynamoDB. Es wurde zuerst über AWS CloudFormation und zuletzt über DynamoDB aufgerufen.

aws: CalledVia Zuerst

Verwenden Sie diesen Schlüssel, um die Services in der Richtlinie mit dem ersten Service zu vergleichen, der im Auftrag des IAM-Auftraggebers (Benutzer oder Rolle) eine Anforderung ausgegeben hat. Weitere Informationen finden Sie unter `aws:CalledVia`.

- Verfügbarkeit – Dieser Schlüssel ist in der Anforderung vorhanden, wenn ein Service die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um mindestens eine andere Anforderung an einen anderen Service auszugeben. Dieser Schlüssel ist nicht vorhanden, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen. Dieser Schlüssel ist auch nicht vorhanden, wenn der Auftraggeber den Aufruf direkt durchführt.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

aws: CalledVia Zuletzt

Verwenden Sie diesen Schlüssel, um die Services in der Richtlinie mit den letzten Services zu vergleichen, die im Auftrag des IAM-Auftraggebers (Benutzer oder Rolle) eine Anforderung ausgegeben haben. Weitere Informationen finden Sie unter `aws:CalledVia`.

- Verfügbarkeit – Dieser Schlüssel ist in der Anforderung vorhanden, wenn ein Service die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um mindestens eine andere Anforderung an einen anderen Service auszugeben. Dieser Schlüssel ist nicht vorhanden, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen. Dieser Schlüssel ist auch nicht vorhanden, wenn der Auftraggeber den Aufruf direkt durchführt.

- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

AWS: via AWSService

Verwenden Sie diesen Schlüssel, um zu überprüfen, ob ein AWS Dienst in Ihrem Namen eine Anfrage an einen anderen Dienst stellt.

Der Anforderungskontextschlüssel gibt `true` zurück, wenn ein Service die Anmeldeinformationen eines IAM-Auftraggebers verwendet, um eine Anforderung im Auftrag des Auftraggebers zu stellen. Der Kontextschlüssel gibt `false` zurück, wenn der Service eine Servicerolle oder eine serviceverknüpfte Rolle verwendet, um einen Aufruf im Auftrag des Auftraggebers durchzuführen. Der Anforderungskontextschlüssel gibt auch `false` zurück, wenn der Auftraggeber den Aufruf direkt durchführt.

- Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext enthalten.
- Datentyp — Boolean
- Werttyp - Einzelwertig

Sie können diesen Bedingungsschlüssel verwenden, um den Zugriff basierend darauf zuzulassen oder zu verweigern, ob eine Anforderung von einem Service ausgegeben wurde.

war: CurrentTime

Verwenden Sie diesen Schlüssel, um das Datum und die Uhrzeit der Anforderung mit dem Datum und der Uhrzeit zu vergleichen, das bzw. die Sie in der Richtlinie angeben. Informationen zum Anzeigen einer Beispielrichtlinie, die diesen Bedingungsschlüssel verwendet, finden Sie unter AWS: Ermöglicht Zugriff basierend auf Datum und Uhrzeit.

- -Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext enthalten.
- Datentyp — Datum
- Werttyp - Einzelwertig

als: EpochTime

Verwenden Sie diesen Schlüssel, um das Datum und die Uhrzeit der Anforderung in Epochen- oder Unix-Zeit mit dem Wert zu vergleichen, den Sie in der Richtlinie angeben. Dieser Schlüssel akzeptiert auch die Anzahl der Sekunden, die seit dem 1. Januar 1970 verstrichen sind.

- Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext enthalten.
- Datentyp — Datum, Numerisch
- Werttyp - Einzelwertig

aws:referer

Verwenden Sie diesen Schlüssel, um den Referrer, der im Client-Browser auf die Anforderung verwiesen hat, mit dem Referrer zu vergleichen, den Sie in der Richtlinie angeben. Der `aws:referer`-Anforderungskontextwert wird vom Aufrufer in einem HTTP-Header bereitgestellt. Der `Referer`-Header ist in einer Webbrowser-Anforderung enthalten, wenn Sie einen Link auf einer Webseite auswählen. Der `Referer`-Header enthält die URL der Webseite, auf der der Link ausgewählt wurde.

- Verfügbarkeit — Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn die Anfrage an die AWS Ressource durch einen Link von einer Webseiten-URL im Browser aufgerufen wurde. Dieser Schlüssel ist nicht für programmatische Anforderungen enthalten, da er keinen Browserlink verwendet, um auf die AWS -Ressource zuzugreifen.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Sie können beispielsweise direkt über eine URL oder einen direkten API-Aufruf auf ein Amazon S3-Objekt zugreifen. Weitere Informationen finden Sie unter Amazon S3-API-Vorgänge direkt mit einem Webbrowser. Wenn Sie von einer URL, die in einer Webseite existiert, auf ein Amazon S3-Objekt zugreifen, wird die URL der Quellwebseite in `aws:referer` verwendet. Wenn Sie auf ein Amazon S3-Objekt zugreifen, indem Sie die URL in Ihren Browser eingeben, ist `aws:referer` nicht vorhanden. Wenn Sie die API direkt aufrufen, ist `aws:referer` auch nicht vorhanden. Sie können den `aws:referer`-Bedingungsschlüssel in einer Richtlinie verwenden, um Anforderungen von einem bestimmten Referer zuzulassen, z. B. einen Link auf einer Webseite in der Domain Ihres Unternehmens.

> **Warning**
>
> Dieser Schlüssel sollte mit Vorsicht verwendet werden. Ein öffentlich bekannter Referer-Header-Wert sollte möglichst nicht eingeschlossen werden. Nicht autorisierte Parteien können mit modifizierten oder benutzerdefinierten Browsern einen beliebigen `aws:referer`-Wert ihrer Wahl bereitstellen. Daher `aws:referer` sollte es nicht verwendet werden, um zu verhindern, dass Unbefugte direkte AWS Anfragen stellen. Die Funktion wird nur bereitgestellt, damit Kunden ihre digitalen, in Amazon S3 gespeicherten Inhalte vor der Referenzierung auf nicht autorisierte Drittanbieter-Websites schützen können.

als: RequestedRegion

Verwenden Sie diesen Schlüssel, um die AWS Region, die in der Anfrage aufgerufen wurde, mit der Region zu vergleichen, die Sie in der Richtlinie angeben. Mit diesem globalen Bedingungsschlüssel können Sie steuern, welche Regionen angefordert werden können. Informationen zur Anzeige der AWS Regionen für die einzelnen Dienste finden Sie unter Dienstendpunkte und Kontingente in der Allgemeine Amazon Web Services-Referenz.

- -Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext enthalten.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Globale Services wie IAM haben einen einzelnen Endpunkt. Da sich dieser Endpunkt in der US East (N. Virginia)-Region befindet, werden IAM-Anrufe immer an die us-east-1 Region gesendet. Wenn Sie beispielsweise eine Richtlinie erstellen, die den Zugriff auf alle Services verweigert, wenn die angeforderte Region nicht us-west-2 ist, schlagen die IAM-Aufrufe immer fehl. Ein Beispiel dafür, wie Sie dieses Problem umgehen können, finden Sie unter NotAction with Deny.

> **Note**
>
> Mit dem Bedingungsschlüssel `aws:RequestedRegion` können Sie steuern, welche Endpunkt eines Services aufgerufen wird, jedoch nicht die Auswirkungen der Operation. Einige Services haben regionsübergreifende Auswirkungen.
> Amazon S3 verfügt beispielsweise über API-Operationen, die sich über Regionen erstrecken.

- Sie können `s3:PutBucketReplication` in einer Region aufrufen (betroffen über den Bedingungsschlüssel `aws:RequestedRegion`), während andere Regionen basierend auf den Konfigurationseinstellungen der Replikation betroffen sind.
- Sie können `s3:CreateBucket` aufrufen, um einen Bucket in einer anderen Region zu erstellen, und mit dem `s3:LocationConstraint`-Bedingungsschlüssel die entsprechenden Regionen steuern.

Sie können diesen Kontextschlüssel verwenden, um den Zugriff auf AWS Dienste innerhalb einer bestimmten Gruppe von Regionen einzuschränken. Mit der folgenden Richtlinie wird Benutzern beispielsweise erlaubt, alle Amazon EC2-Instances in der AWS Management Console anzuzeigen. Sie gestattet den Benutzern aber nur Änderungen an Instances in Irland (eu-west-1), London (eu-west-2) und Paris (eu-west-3).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
```

```
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

RequestTagaws/Tag-Schlüssel

Verwenden Sie diesen Schlüssel, um das Tag-Schlüssel-Wert-Paar, das in der Anforderung übergeben wurde, mit dem Tag-Paar zu vergleichen, das Sie in der Richtlinie angeben. Sie können beispielsweise prüfen, ob die Anforderung den Tag-Schlüssel `"Dept"` enthält und dieser den Wert `"Accounting"` hat. Weitere Informationen finden Sie unter Zugriffssteuerung während AWS - Anforderungen.

- Verfügbarkeit – Dieser Schlüssel ist im Anforderungskontext enthalten, wenn Tag-Schlüssel-Wert-Paare in der Anforderung übergeben werden. Wenn mehrere Tags in der Anforderung übergeben werden, gibt es einen Kontextschlüssel für jedes Tag-Schlüssel-Wert-Paar.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

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. Bei Tag-Schlüsseln und -Werten wird nicht zwischen Groß- und Kleinschreibung unterschieden. Dies bedeutet Folgendes: Wenn Sie `"aws:RequestTag/TagKey1": "Value1"` im Bedingungselement Ihrer Richtlinie angeben, stimmt die Bedingung mit einem Ressourcen-Tag-Schlüssel mit dem Namen `TagKey1` oder `tagkey1` überein, aber nicht mit beiden.

Dieses Beispiel zeigt, dass der Schlüssel zwar einwertig ist, Sie aber dennoch mehrere Schlüssel-Wert-Paare in einer Anforderung verwenden können, wenn die Schlüssel unterschiedlich sind.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:::instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

als: TagKeys

Verwenden Sie diesen Schlüssel, um die Tag-Schlüssel in einer Anforderung mit den Schlüsseln zu vergleichen, die Sie in der Richtlinie angeben. Wir empfehlen, dass Sie bei der Verwendung von Richtlinien zur Zugriffskontrolle mithilfe von Tags den `aws:TagKeys`-Bedingungsschlüssel verwenden, um festzulegen, welche Tag-Schlüssel zulässig sind. Beispielrichtlinien und weitere Informationen finden Sie unter the section called “Zugriffssteuerung auf der Grundlage von Tag-Schlüsseln”.

- Verfügbarkeit – Dieser Schlüssel ist in den Anforderungskontext enthalten, wenn die Operation die Übergabe von Tags in der Anforderung unterstützt.
- Datentyp — Zeichenfolge (Liste)
- Werttyp - Mehrwertig

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"]`).

Da Sie mehrere Tag-Schlüssel-Wert-Paare in eine Anforderung aufnehmen können, könnte der Anforderungsinhalt eine mehrwertige Anforderung sein. In diesem Fall müssen Sie die Operatoren

`ForAllValues` oder `ForAnyValue` verwenden. Weitere Informationen finden Sie unter Mehrwertige Kontextschlüssel.

Einige Services unterstützen das Markieren mit Ressourcenoperationen, wie etwa das Erstellen, Ändern oder Löschen einer Ressource. Um das Markieren und Operationen als einzelnen Aufruf zu erlauben, müssen Sie eine Richtlinie erstellen, die die Aktionen Markieren und Ressourcenbearbeitung enthält. Anschließend können Sie den `aws:TagKeys`-Bedingungsschlüssel zum Erzwingen mit spezifischen Tag-Schlüssel in der Anforderung verwenden. Beispiel: Zum Einschränken von Tags, wenn jemand einen Amazon EC2-Snapshot erstellt, müssen Sie die Aktion zum Erstellen eines `ec2:CreateSnapshot` und die Aktion zum Markieren von `ec2:CreateTags` in die Richtlinie einbeziehen. Eine Richtlinie für dieses Szenario, das verwendet`aws:TagKeys`, finden Sie unter Erstellen eines Snapshots mit Tags im Amazon EC2 EC2-Benutzerhandbuch.

aws: SecureTransport

Mit diesem Schlüssel können Sie überprüfen, ob die Anforderung über SSL gesendet wurde. Der Anforderungskontext gibt `true` oder `false` zurück. In einer Richtlinie können Sie bestimmte Aktionen nur zulassen, wenn die Anforderung mit SSL gesendet wird.

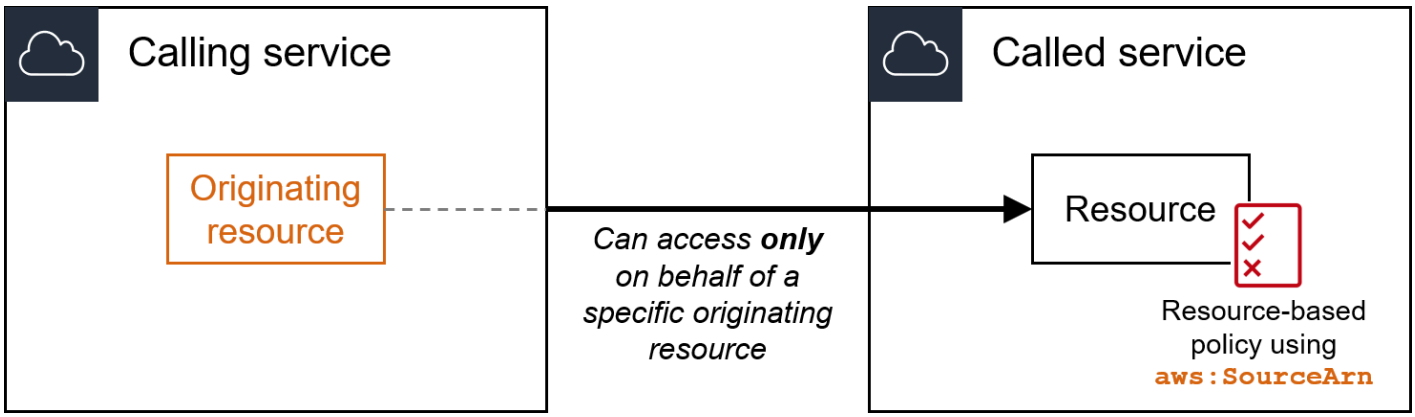
- Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext enthalten.
- Datentyp — Boolean
- Werttyp - Einzelwertig

war: SourceArn

Verwenden Sie diesen Schlüssel, um den Amazon-Ressourcennamen (ARN) der Ressource, die eine service-to-service Anfrage stellt, mit dem ARN zu vergleichen, den Sie in der Richtlinie angeben, aber nur, wenn die Anfrage von einem AWS Service Principal gestellt wird. Wenn der ARN der Quelle die Konto-ID enthält, ist es nicht erforderlich, `aws:SourceAccount` mit `aws:SourceArn` zu verwenden.

Dieser Schlüssel funktioniert nicht mit dem ARN des Auftraggebers, der die Anforderung stellt. Nutzen Sie stattdessen `war: PrincipalArn`.

- Verfügbarkeit — Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Aufruf Ihrer Ressource direkt von einem AWS Service Principal im Namen einer Ressource erfolgt, für die die Konfiguration die service-to-service Anfrage ausgelöst hat. Der aufrufende Service muss den ARN der ursprünglichen Ressource an den aufgerufenen Service übergeben.



Die folgenden Service-Integrationen unterstützen diesen globalen Bedingungsschlüssel nicht:

Aufrufender Service (Service-Prinzipal)	Aufgerufener Service (ressourcenbasierte Richtlinie)	Beschreibung
logdelivery.elb.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket
logdelivery.elasticloadbalancing.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket

> **Note**
>
> Nicht alle Serviceintegrationen mit AWS Security Token Service (AWS STS) und AWS Key Management Service (AWS KMS) werden unterstützt. Weitere Informationen finden Sie in der Dokumentation des aufrufenden Service. Die Verwendung von `aws:SourceArn` KMS-Schlüsselrichtlinien für Schlüssel, die AWS-Services über KMS-Schlüsselzuweisungen verwendet werden, kann zu unerwartetem Verhalten führen.

- Datentyp — ARN, Zeichenfolge

AWS empfiehlt, beim Vergleich von ARNs ARN-Operatoren anstelle von Zeichenkettenoperatoren zu verwenden.

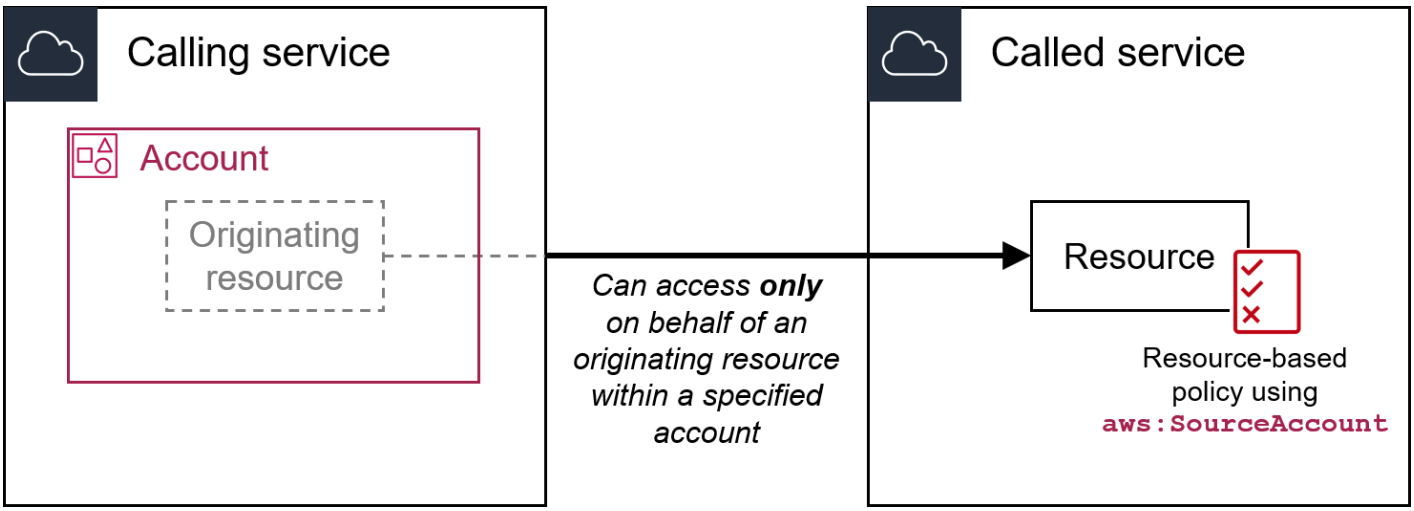
- Werttyp - Einzelwertig

Sie können diesen Bedingungsschlüssel verwenden, um zu verhindern, dass ein AWS Dienst bei Transaktionen zwischen Diensten als verwirrter Stellvertreter verwendet wird. Verwenden Sie diesen Schlüssel nur in ressourcenbasierten Richtlinien, bei denen es sich um einen `Principal` Prinzipal handelt AWS-Service . Legen Sie den Wert dieses Bedingungsschlüssels auf den ARN der Ressource in der Anforderung fest. Wenn beispielsweise eine Aktualisierung eines Amazon-S3-Buckets eine Amazon-SNS-Themenveröffentlichung auslöst, ruft der Amazon-S3-Service den `sns:Publish`-API-Vorgang auf. Setzen Sie in der Themenrichtlinie, die den Vorgang `sns:Publish` erlaubt, den Wert des Bedingungsschlüssels auf den ARN des Amazon-S3-Buckets. Informationen darüber, wie und wann dieser Bedingungsschlüssel empfohlen wird, finden Sie in der Dokumentation der AWS Dienste, die Sie verwenden.

aws: SourceAccount

Verwenden Sie diesen Schlüssel, um die Konto-ID der Ressource, die eine service-to-service Anfrage stellt, mit der Konto-ID zu vergleichen, die Sie in der Richtlinie angeben. Dies gilt jedoch nur, wenn die Anfrage von einem AWS Service Principal gestellt wird.

- Verfügbarkeit — Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Aufruf Ihrer Ressource direkt von einem AWS Service Principal im Namen einer Ressource erfolgt, für die die Konfiguration die service-to-service Anfrage ausgelöst hat. Der aufrufende Service muss die Konto-ID der ursprünglichen Ressource an den aufgerufenen Service weitergeben.



Die folgenden Service-Integrationen unterstützen diesen globalen Bedingungsschlüssel nicht:

Aufrufender Service (Service-Prinzipal)	Aufgerufener Service (ressourcenbasierte Richtlinie)	Beschreibung
logdelivery.elb.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket
logdelivery.elasticloadbalancing.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket

> **Note**
>
> Nicht alle Serviceintegrationen mit AWS Security Token Service (AWS STS) und AWS Key Management Service (AWS KMS) werden unterstützt. Weitere Informationen finden Sie in der Dokumentation des aufrufenden Service. Die Verwendung von `aws:SourceAccount` KMS-Schlüsselrichtlinien für Schlüssel, die AWS-Services über KMS-Schlüsselzuweisungen verwendet werden, kann zu unerwartetem Verhalten führen.

- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

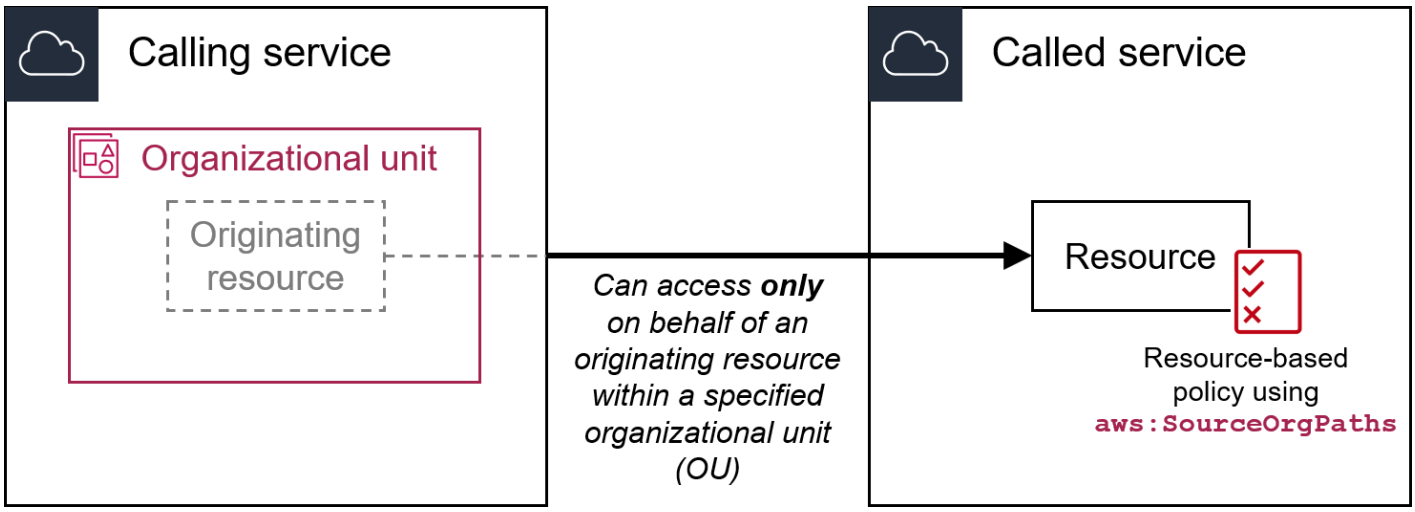
Sie können diesen Bedingungsschlüssel verwenden, um zu verhindern, dass ein AWS Dienst bei Transaktionen zwischen Diensten als verwirrter Stellvertreter verwendet wird. Verwenden Sie diesen Schlüssel nur in ressourcenbasierten Richtlinien, bei denen es sich um einen `Principal` Prinzipal handelt AWS-Service . Legen Sie den Wert dieses Bedingungsschlüssels auf die Konto-ID der Ressource in der Anforderung fest. Wenn beispielsweise eine Aktualisierung eines Amazon-S3-Buckets eine Amazon-SNS-Themenveröffentlichung auslöst, ruft der Amazon-S3-Service den `sns:Publish`-API-Vorgang auf. Setzen Sie in der Themenrichtlinie, die den Vorgang `sns:Publish` erlaubt, den Wert des Bedingungsschlüssels auf die Konto-ID des Amazon-S3-

Buckets. Informationen darüber, wie und wann dieser Bedingungsschlüssel empfohlen wird, finden Sie in der Dokumentation der AWS Dienste, die Sie verwenden.

aws: SourceOrg Pfade

Verwenden Sie diesen Schlüssel, um den AWS Organizations Pfad der Ressource, die eine service-to-service Anfrage stellt, mit dem Organisationspfad zu vergleichen, den Sie in der Richtlinie angeben. Dies gilt jedoch nur, wenn die Anfrage von einem AWS Dienstprinzipal gestellt wird. Ein -Organizations-Pfad ist eine Textdarstellung der Struktur einer Organizations-Entität. Weitere Informationen zu Pfaden und deren Verwendung finden Sie unter Den AWS Organizations - Entitätspfad verstehen.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Anruf an Ihre Ressource direkt von einem AWS -Service-Prinzipal im Namen einer Ressource erfolgt, die einem Konto gehört, das Mitglied einer Organisation ist. Der aufrufende Service übergibt den Organisations-Pfad der ursprünglichen Ressource an den aufgerufenen Service.



Die folgenden Service-Integrationen unterstützen diesen globalen Bedingungsschlüssel nicht:

Aufrufender Service (Service-Prinzipal)	Aufgerufener Service (ressourcenbasierte Richtlinie)	Beschreibung
logdelivery.elb.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket

Aufrufender Service (Service-Prinzipal)	Aufgerufener Service (ressourcenbasierte Richtlinie)	Beschreibung
logdelivery.elasticloadbalancing.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket
Alle Service-Prinzipale	Amazon-Lex-Bot	Erlaube AWS-Services die Verwendung des Amazon Lex Lex-Bot

> **Note**
>
> Nicht alle Serviceintegrationen mit AWS Security Token Service (AWS STS) und AWS Key Management Service (AWS KMS) werden unterstützt. Weitere Informationen finden Sie in der Dokumentation des aufrufenden Service. Die Verwendung von `aws:SourceOrgPaths` KMS-Schlüsselrichtlinien für Schlüssel, die AWS-Services über KMS-Schlüsselzuweisungen verwendet werden, kann zu unerwartetem Verhalten führen.

- Datentyp — Zeichenfolge (Liste)
- Werttyp - Mehrwertig

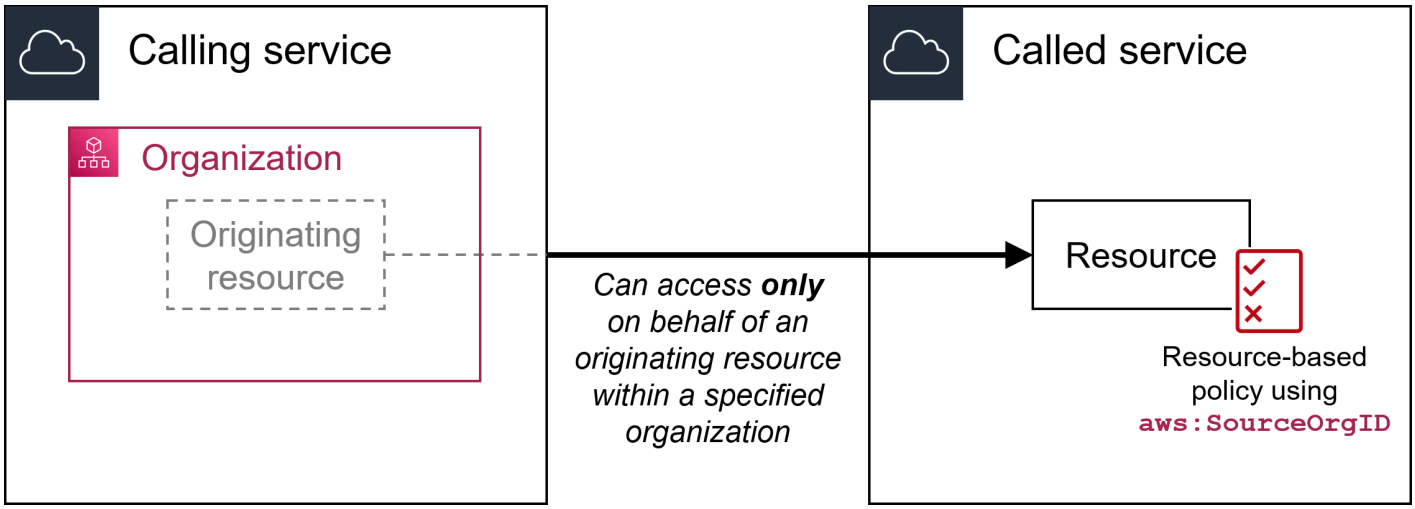
Sie können diesen Bedingungsschlüssel verwenden, um zu verhindern, dass ein AWS Dienst bei Transaktionen zwischen Diensten als verwirrter Stellvertreter verwendet wird. Verwenden Sie diesen Schlüssel nur in ressourcenbasierten Richtlinien, bei denen es sich um einen `Principal` Prinzipal handelt AWS-Service . Legen Sie den Wert dieses Bedingungsschlüssels auf den Organisations-Pfad der Ressource in der Anforderung fest. Wenn beispielsweise eine Aktualisierung eines Amazon-S3-Buckets eine Amazon-SNS-Themenveröffentlichung auslöst, ruft der Amazon-S3-Service den `sns:Publish`-API-Vorgang auf. Setzen Sie in der Themenrichtlinie, die den Vorgang `sns:Publish` zulässt, den Wert des Bedingungsschlüssels auf den Organisations-Pfad des Amazon-S3-Buckets. Informationen darüber, wie und wann dieser Bedingungsschlüssel empfohlen wird, finden Sie in der Dokumentation der AWS Dienste, die Sie verwenden.

`aws:SourceOrgPaths` ist ein mehrwertiger Bedingungsschlüssel. Mehrwertige Bedingungsschlüssel können im Anforderungskontext mehrere Werte haben. Sie müssen die Set-Operatoren `ForAnyValue` oder `ForAllValues` zusammen mit String-Bedingungsoperatoren für diesen Schlüssel verwenden. Weitere Hinweise zu mehrwertigen Bedingungsschlüsseln finden Sie unter Mehrwertige Kontextschlüssel.

aws: SourceOrg ID

Verwenden Sie diesen Schlüssel, um die Organisations-ID der Ressource, die eine service-to-service Anfrage stellt, mit der Organisations-ID zu vergleichen, die Sie in der Richtlinie angeben. Dies gilt jedoch nur, wenn die Anfrage von einem AWS Dienstprinzipal gestellt wird. Wenn Sie Konten in einer Organisation in AWS Organizations hinzufügen und entfernen, schließen Richtlinien, die den Schlüssel `aws:SourceOrgID` enthalten, automatisch die richtigen Konten ein und Sie müssen die Richtlinien nicht manuell aktualisieren.

- Verfügbarkeit – Dieser Schlüssel ist nur dann im Anforderungskontext enthalten, wenn der Anruf an Ihre Ressource direkt von einem AWS -Service-Prinzipal im Namen einer Ressource erfolgt, die einem Konto gehört, das Mitglied einer Organisation ist. Der aufrufende Service übergibt den ARN der ursprünglichen Ressource an den aufgerufenen Service.



Die folgenden Service-Integrationen unterstützen diesen globalen Bedingungsschlüssel nicht:

Aufrufender Service (Service-Prinzipal)	Aufgerufener Service (ressourcenbasierte Richtlinie)	Beschreibung
logdelivery.elb.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket
logdelivery.elasticloadbalancing.amazonaws.com	Amazon-S3-Bucket	Aktivieren Sie die Elastic-Load-Balancing-Zugriffsprotokollierung im Amazon-S3-Bucket
Alle Service-Prinzipale	Amazon-Lex-Bot	Erlaube AWS-Services die Verwendung des Amazon Lex Lex-Bot

> **Note**
>
> Nicht alle Serviceintegrationen mit AWS Security Token Service (AWS STS) und AWS Key Management Service (AWS KMS) werden unterstützt. Weitere Informationen finden Sie in der Dokumentation des aufrufenden Service. Die Verwendung von `aws:SourceOrgID` KMS-Schlüsselrichtlinien für Schlüssel, die AWS-Services über KMS-Schlüsselzuweisungen verwendet werden, kann zu unerwartetem Verhalten führen.

- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

Sie können diesen Bedingungsschlüssel verwenden, um zu verhindern, dass ein AWS Dienst bei Transaktionen zwischen Diensten als verwirrter Stellvertreter verwendet wird. Verwenden Sie diesen Schlüssel nur in ressourcenbasierten Richtlinien, bei denen es sich um einen `Principal` Prinzipal handelt AWS-Service . Legen Sie den Wert dieses Bedingungsschlüssels auf die Organisations-ID der Ressource in der Anforderung fest. Wenn beispielsweise eine Aktualisierung eines Amazon-S3-Buckets eine Amazon-SNS-Themenveröffentlichung auslöst, ruft der Amazon-S3-Service den

`sns:Publish`-API-Vorgang auf. Setzen Sie in der Themenrichtlinie, die den Vorgang `sns:Publish` erlaubt, den Wert des Bedingungsschlüssels auf die Organisations-ID des Amazon-S3-Buckets. Informationen darüber, wie und wann dieser Bedingungsschlüssel empfohlen wird, finden Sie in der Dokumentation der AWS Dienste, die Sie verwenden.

aws: UserAgent

Verwenden Sie diesen Schlüssel, um die Client-Anwendung des Anforderers mit der Anwendung zu vergleichen, die Sie in der Richtlinie angeben.

- Verfügbarkeit – Dieser Schlüssel ist immer im Anforderungskontext enthalten.
- Datentyp — Zeichenfolge
- Werttyp - Einzelwertig

> **Warning**
>
> Dieser Schlüssel sollte mit Vorsicht verwendet werden. Da der Wert `aws:UserAgent` vom Aufrufer in einem HTTP-Header bereitgestellt wird, können nicht autorisierte Parteien modifizierte oder benutzerdefinierte Browser verwenden, um beliebige `aws:UserAgent`-Werte bereitzustellen. Daher `aws:UserAgent` sollte es nicht verwendet werden, um zu verhindern, dass Unbefugte direkte AWS Anfragen stellen. Sie können es verwenden, um nur bestimmte Client-Anwendungen zuzulassen, und zwar erst nach dem Testen Ihrer Richtlinie.

Andere dienstübergreifende Bedingungsschlüssel

AWS STS unterstützt SAML-basierte Verbundbedingungsschlüssel und dienstübergreifende Bedingungsschlüssel für den OIDC-Verbund. Diese Schlüssel sind verfügbar, wenn ein Benutzer, der mithilfe von SAML verbunden war, Operationen in anderen Diensten ausführt. AWS

IAM- und AWS STS Bedingungskontextschlüssel

Sie können das `Condition` Element in einer JSON-Richtlinie verwenden, um den Wert von Schlüsseln zu testen, die im Anforderungskontext aller AWS Anfragen enthalten sind. Diese Schlüssel enthalten Informationen zu der Anforderung selbst oder zu den Ressourcen, die die Anforderung referenziert. Sie können die Schlüssel auf bestimmte Werte überprüfen, bevor Sie die angeforderte Aktion zulassen. So können Sie genau steuern, wann Ihre JSON-Richtlinienanweisungen mit einer eingehenden Anforderung übereinstimmen. Weitere Informationen

zur Verwendung des Elements `Condition` in JSON-Richtlinien finden Sie unter IAM-JSON-Richtlinienelemente: Condition.

In diesem Thema werden die Schlüssel beschrieben, die vom IAM-Dienst (mit einem `iam:` Präfix) und dem Dienst () AWS Security Token Service (mit einem `sts:` Präfix AWS STS) definiert und bereitgestellt werden. Verschiedene andere AWS Dienste stellen auch dienstspezifische Schlüssel bereit, die für die von diesem Dienst definierten Aktionen und Ressourcen relevant sind. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste. Weitere Informationen zu den unterstützten Bedingungsschlüsseln können Sie meist der Dokumentation des jeweiligen Service entnehmen. Weitere Informationen zu Schlüsseln, die Sie in Richtlinien für Amazon-S3-Ressourcen verwenden können, finden Sie beispielsweise unter Amazon-S3-Richtlinienschlüssel im Benutzerhandbuch für Amazon Simple Storage Service.

Themen

Verfügbare Schlüssel für IAM

Sie können die folgenden Bedingungsschlüssel in Richtlinien verwenden, die zur Zugriffssteuerung auf IAM-Ressourcen genutzt werden:

Ich bin: AssociatedResourceArn

Funktioniert mit ARN-Operatoren.

Gibt den ARN der Ressource an, der diese Rolle beim Zielservice zugeordnet wird. Die Ressource gehört normalerweise zu dem Service, an den der Auftraggeber die Rolle übergibt. Manchmal kann die Ressource zu einem dritten Service gehören. Sie können beispielsweise eine Rolle an Amazon EC2 Auto Scaling übergeben, die sie auf einer Amazon EC2-Instance verwenden. In diesem Fall würde die Bedingung mit dem ARN der Amazon EC2-Instance übereinstimmen.

Dieser Bedingungsschlüssel gilt nur für die PassRoleAktion in einer Richtlinie. Er kann nicht verwendet werden, um eine andere Aktion zu beschränken.

Verwenden Sie diesen Bedingungsschlüssel in einer Richtlinie, um einer Entität das Übergeben einer Rolle zu ermöglichen, jedoch nur, wenn diese Rolle der angegebenen Ressource zugeordnet ist. Sie können Platzhalter (*) verwenden, um Vorgänge zu ermöglichen, die für einen bestimmten Ressourcentyp ausgeführt werden, ohne die Regions- oder Ressourcen-ID einzuschränken. Beispielsweise können Sie einem IAM-Benutzer`us-east-1` oder einer entsprechenden Rolle erlauben, eine beliebige Rolle an den Amazon EC2-Service zu übergeben, die mit Instances in der Region oder `us-west-1` verwendet werden soll. Der IAM-Benutzer oder die IAM-Rolle wäre nicht berechtigt, Rollen an andere Services zu übergeben. Darüber hinaus ermöglicht Amazon EC2 nicht, die Rolle mit Instances in anderen Regionen zu verwenden.

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-west-1:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> AWS Dienste, die iam: unterstützen PassedToService auch diesen Bedingungsschlüssel.

Ich bin: AWSServiceName

Funktioniert mit Zeichenfolgenoperatoren.

Gibt den AWS Dienst an, dem diese Rolle zugewiesen ist.

In diesem Beispiel erlauben Sie einer Entität das Erstellen einer serviceverknüpften Rolle, wenn der Servicename access-analyzer.amazonaws.com. lautet.

```
{
```

```
   "Version": "2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
     }]
 }
```

iam:FIDO-certification

Funktioniert mit Zeichenfolgenoperatoren.

Überprüft die FIDO-Zertifizierungsstufe des MFA-Geräts zum Zeitpunkt der Registrierung eines FIDO-Sicherheitsschlüssels. Die Gerätezertifizierung wird vom FIDO Alliance Metadata Service (MDS) abgerufen. Wenn sich der Zertifizierungsstatus oder die Stufe Ihres FIDO-Sicherheitsschlüssels ändert, wird dieser nicht aktualisiert, es sei denn, das Gerät wird deregistriert und erneut registriert, um die aktualisierten Zertifizierungsinformationen abzurufen.

Mögliche Werte von L1, L1plus, L2, L2plus, L3, L3plus

In diesem Beispiel registrieren Sie einen Sicherheitsschlüssel und erhalten die FIDO Level 1 Plus-Zertifizierung für Ihr Gerät.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-2-certification

Funktioniert mit Zeichenfolgenoperatoren.

Überprüft die FIPS-140-2-Validierungszertifizierungsstufe des MFA-Geräts zum Zeitpunkt der Registrierung eines FIDO-Sicherheitsschlüssels. Die Gerätezertifizierung wird vom FIDO Alliance Metadata Service (MDS) abgerufen. Wenn sich der Zertifizierungsstatus oder die Stufe Ihres FIDO-Sicherheitsschlüssels ändert, wird dieser nicht aktualisiert, es sei denn, das Gerät wird deregistriert und erneut registriert, um die aktualisierten Zertifizierungsinformationen abzurufen.

Mögliche Werte von L1, L2, L3, L4

In diesem Beispiel registrieren Sie einen Sicherheitsschlüssel und erhalten die Zertifizierung FIPS-140-2 Level 2 für Ihr Gerät.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-3-certification

Funktioniert mit Zeichenfolgenoperatoren.

Überprüft die FIPS-140-3-Validierungszertifizierungsstufe des MFA-Geräts zum Zeitpunkt der Registrierung eines FIDO-Sicherheitsschlüssels. Die Gerätezertifizierung wird vom FIDO Alliance Metadata Service (MDS) abgerufen. Wenn sich der Zertifizierungsstatus oder die Stufe Ihres FIDO-Sicherheitsschlüssels ändert, wird dieser nicht aktualisiert, es sei denn, das Gerät wird deregistriert und erneut registriert, um die aktualisierten Zertifizierungsinformationen abzurufen.

Mögliche Werte von L1, L2, L3, L4

In diesem Beispiel registrieren Sie einen Sicherheitsschlüssel und erhalten die Zertifizierung FIPS-140-3 Level 3 für Ihr Gerät.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
            }
        }
    ]

 }
```

ich bin: RegisterSecurityKey

Funktioniert mit Zeichenfolgenoperatoren.

Prüft den aktuellen Status der MFA-Geräteaktivierung.

Mögliche Werte von `Create` oder `Activate`.

In diesem Beispiel registrieren Sie einen Sicherheitsschlüssel und erhalten die Zertifizierung FIPS-140-3 Level 1 für Ihr Gerät.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
```

```
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

}
```

ich bin: OrganizationsPolicyId

Funktioniert mit Zeichenfolgenoperatoren.

Überprüft, ob die Richtlinie mit der angegebenen AWS Organizations ID mit der in der Anfrage verwendeten Richtlinie übereinstimmt. Informationen zum Anzeigen einer IAM-Beispielrichtlinie, die diesen Bedingungsschlüssel verwendet, finden Sie unter IAM: Anzeigen von Informationen zum letzten Zugriff auf einen Service für eine Organizations-Richtlinie.

ich bin: PassedToService

Funktioniert mit Zeichenfolgenoperatoren.

Gibt den ServiceAuftraggeber des Services an, an den eine Rolle übergeben werden kann. Dieser Bedingungsschlüssel gilt nur für die PassRoleAktion in einer Richtlinie. Er kann nicht verwendet werden, um eine andere Aktion zu beschränken.

Wenn Sie diesen Bedingungsschlüssel in einer Richtlinie verwenden, geben Sie den Service über einen Dienstauftraggeber an. Ein ServiceAuftraggeber ist der Name eines Services, der im Element `Principal` einer Richtlinie angegeben werden kann. Standardformat: `SERVICE_NAME_URL.amazonaws.com`.

Sie können `iam:PassedToService` verwenden, um die Benutzer einzuschränken, damit sie Rollen nur an bestimmte Services übergeben können. Beispielsweise könnte ein Benutzer eine Servicerolle erstellen, die CloudWatch darauf vertraut, in seinem Namen Protokolldaten in einen Amazon S3 S3-Bucket zu schreiben. Der Benutzer muss dann eine Berechtigungsrichtlinie und eine Vertrauensrichtlinie an die neue Servicerolle anfügen. In diesem Fall muss die Vertrauensrichtlinie `cloudwatch.amazonaws.com` im Element `Principal` angeben. Eine Richtlinie, an die der Benutzer die Rolle übergeben kann CloudWatch, finden Sie unterIAM: Übergeben Sie eine IAM-Rolle an einen bestimmten Dienst AWS.

Mit diesem Bedingungsschlüssel können Sie sicherstellen, dass Benutzer Servicerollen nur für die von Ihnen angegebenen Services erstellen. Wenn beispielsweise ein Benutzer mit der vorherigen

Richtlinie versucht, eine Servicerolle für Amazon EC2 zu erstellen, schlägt der Vorgang fehl. Der Fehlschlag tritt auf, weil der Benutzer nicht über die Berechtigung verfügt, die Rolle an Amazon EC2 zu übergeben.

Manchmal übergeben Sie eine Rolle an einen Service, der die Rolle dann an einen anderen Service übergibt. `iam:PassedToService` enthält nur den endgültigen Dienst, der die Rolle übernimmt, nicht den Zwischendienst, der die Rolle übergibt.

> **Note**
>
> Einige Services unterstützen diesen Bedingungsschlüssel nicht.

ich bin: PermissionsBoundary

Funktioniert mit ARN-Operatoren.

Überprüft, ob die angegebene Richtlinie als Berechtigungsgrenze an die IAM-Auftraggeber-Ressource angefügt. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten

iam:PolicyARN

Funktioniert mit ARN-Operatoren.

Prüft den Amazon-Ressourcennamen (ARN) einer verwalteten Richtlinie in Anforderungen mit Beteiligung einer verwalteten Richtlinie. Weitere Informationen finden Sie unter Steuern des Zugriffs auf Richtlinien.

iam:/Schlüsselname ResourceTag

Funktioniert mit Zeichenfolgenoperatoren.

Überprüft, ob das an die Identitätsressource (Benutzer oder Rolle) angefügte Tag mit dem angegebenen Schlüsselnamen und Wert übereinstimmt.

> **Note**
>
> IAM und AWS STS unterstützen sowohl den `iam:ResourceTag` IAM-Bedingungsschlüssel als auch den globalen Bedingungsschlüssel. `aws:ResourceTag`

Sie können IAM-Ressourcen benutzerdefinierte Attribute in Form eines Schlüssel-Wert-Paares hinzufügen. Weitere Informationen zum Markieren von IAM-Ressourcen finden Sie unter the section called “Markieren von IAM-Ressourcen”. Sie können `ResourceTag` verwenden, um den Zugriff auf AWS -Ressourcen, einschließlich IAM-Ressourcen, zu steuern. IAM unterstützt jedoch keine Tags für Gruppen. Daher können Sie Tags nicht verwenden, um den Zugriff auf Gruppen zu steuern.

Dieses Beispiel zeigt, wie Sie eine identitätsbasierte Richtlinie erstellen könnten, die das Löschen von Benutzern mit dem **`status=terminated`**-Tag erlaubt. Um diese Richtlinie zu verwenden, ersetzen Sie den *`kursiv gedruckten Platzhaltertext`* in der Beispielrichtlinie durch Ihre eigenen Informationen. Befolgen Sie dann die Anweisungen unter Erstellen einer Richtlinie oder Bearbeiten einer Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status": "terminated"}}
    }]
}
```

Verfügbare Schlüssel für AWS den OIDC-Verbund

Sie können den OIDC-Verbund verwenden, um Benutzern, die über einen OpenID Connect-kompatiblen Identitätsanbieter (IdP) authentifiziert wurden, temporäre Sicherheitsanmeldedaten für einen IAM OpenID Connect (OIDC) -Identitätsanbieter in Ihrem Konto zuzuweisen. AWS Beispiele für solche Anbieter sind GitHub Amazon Cognito, Login with Amazon und Google. Identitätstoken und Zugriffstoken von Ihrem eigenen IdP können ebenso verwendet werden wie Dienstkonto-Token, die Amazon Elastic Kubernetes Service Service-Workloads gewährt wurden.

Sie können AWS OIDC-Bedingungskontextschlüssel verwenden, um Richtlinien zu schreiben, die den Zugriff verbundener Benutzer auf Ressourcen beschränken, die einem bestimmten Anbieter, einer bestimmten App oder einem bestimmten Benutzer zugeordnet sind. Diese Schlüssel werden in der Regel in der Vertrauensrichtlinie für eine Rolle verwendet. Definieren Sie Bedingungsschlüssel mithilfe des Namens des OIDC-Anbieters (`token.actions.githubusercontent.com`), gefolgt von einem Anspruch (): `:aud` **`token.actions.githubusercontent.com:aud`**

Einige OIDC-Verbundbedingungsschlüssel können in der Rollensitzung verwendet werden, um den Ressourcenzugriff zu autorisieren. Wenn der Wert in der Spalte In Sitzung verfügbar auf Ja steht, können Sie diese Bedingungsschlüssel in Richtlinien verwenden, um zu definieren, auf welche Benutzer in anderen Diensten zugreifen dürfen. AWS Wenn ein Anspruch in einer Sitzung nicht verfügbar ist, kann der OIDC-Bedingungskontextschlüssel nur in einer Rollenvertrauensrichtlinie für die AssumeRoleWithWebIdentityErstauthentifizierung verwendet werden.

Wählen Sie Ihren IdP aus, um zu sehen, wie Ansprüche Ihres IdP dem IAM-Bedingungskontext zugeordnet werden. AWS

Default

In der Standardeinstellung werden die Standard-OIDC-Ansprüche aufgeführt und wie sie den Bedingungskontextschlüsseln zugeordnet AWS STS werden. AWS Mit diesen Schlüsseln können Sie den Zugriff auf eine Rolle steuern. Vergleichen Sie dazu die AWS STS Bedingungsschlüssel mit den Werten in der Spalte IdP JWT Claim. Verwenden Sie diese Zuordnung, wenn Ihr IdP nicht in den Tab-Optionen aufgeführt ist.

GitHub Actions, Workflows und Google sind einige Beispiele dafür IdPs , die Standardimplementierung in ihrem OIDC JWT-ID-Token verwenden.

AWS STS Bedingungsschlüssel	IdP JWT-Anspruch	Während der Sitzung verfügbar
amr	amr	Ja
aud	azp Wenn kein Wert für angegeben ist`azp`, wird der `aud` Bedingungsschlüssel dem `aud` Anspruch zugeordnet.	Ja
email	email	Nein
oaud	aud	Nein
sub	sub	Ja

Weitere Hinweise zur Verwendung von OIDC-Bedingungskontextschlüsseln mit GitHub finden Sie unter. Konfiguration einer Rolle für den GitHub OIDC-Identitätsanbieter Weitere Informationen zu den Google-Feldern `aud` und `azp` finden Sie im Google Identity Platform OpenID Connect-Leitfaden.

amr

Funktioniert mit Zeichenfolgenoperatoren. Der Schlüssel enthält mehrere Werte, d. h. Sie überprüfen ihn in einer Richtlinie mithilfe von Bedingungsmengenoperatoren.

Beispiel: `token.actions.githubusercontent.com:amr`

Die Referenz zu Authentifizierungsmethoden enthält Anmeldeinformationen über den Benutzer. Der Schlüssel kann die folgenden Werte enthalten:

- Wenn der Benutzer nicht authentifiziert wurde, enthält der Schlüssel nur `unauthenticated`.
- Wenn der Benutzer authentifiziert ist, enthält der Schlüssel den Wert `authenticated` und den Namen des Anmeldeanbieters, der im Call (`accounts.google.com`) verwendet wurde.

aud

Funktioniert mit Zeichenfolgenoperatoren.

Beispiele:

- `accounts.google.com:aud`
- `token.actions.githubusercontent.com:aud`

Verwenden Sie den `aud` Bedingungsschlüssel, um zu überprüfen, ob die Zielgruppe der Zielgruppe entspricht, die Sie in der Richtlinie angegeben haben. Sie können den AUD-Schlüssel zusammen mit dem Unterschlüssel für denselben Identitätsanbieter verwenden.

Dieser Bedingungsschlüssel wird aus den folgenden Token-Feldern festgelegt:

- `aud` for OAuth 2.0-Google-Client-IDs Ihrer Anwendung, wenn das Feld `azp` nicht festgelegt ist. Wenn das `azp`-Feld festgelegt ist, stimmt das `aud`-Feld mit dem Bedingungsschlüssel `accounts.google.com:oaud` überein.
- `azp`, wenn das Feld `azp` festgelegt ist. Dies kann bei hybriden Anwendungen passieren, bei denen eine Webanwendung und eine Android-Anwendung eine unterschiedliche OAuth 2.0 Google-Client-ID haben, aber dasselbe Google-APIs-Projekt verwenden.

Wenn Sie eine Richtlinie mit dem `accounts.google.com:aud`-Bedingungsschlüssel schreiben, müssen Sie wissen, ob die Anwendung eine Hybridanwendung ist, die das Feld `azp` festlegt.

`azp`-Feld nicht festgelegt

Die folgende Beispielrichtlinie funktioniert für nicht-hybride Anwendungen, die das Feld `azp` nicht festlegen. In diesem Fall stimmt der Wert des Google-ID-Token `aud`-Felds mit dem `accounts.google.com:aud`- und dem `accounts.google.com:oaud`-Bedingungsschlüsselwert überein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

`azp`-Feld festgelegt

Die folgende Beispielrichtlinie funktioniert für hybride Anwendungen, die das Feld `azp` festlegen. In diesem Fall stimmt der Wert des Google-ID-Token `aud`-Felds nur mit dem Wert des Bedingungsschlüssels `accounts.google.com:oaud` überein. Der Wert des Feldes `azp` entspricht dem Wert des `accounts.google.com:aud`-Bedingungsschlüssels.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
```

```
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

E-Mail

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel: `accounts.google.com:email`

Dieser Bedingungsschlüssel validiert die E-Mail-Adresse des Benutzers. Der Wert dieses Antrags ist möglicherweise nicht eindeutig auf dieses Konto beschränkt und kann sich im Laufe der Zeit ändern. Daher sollten Sie diesen Wert nicht als primäre Kennung zur Überprüfung Ihres Benutzerdatensatzes verwenden.

oaud

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel: `accounts.google.com:oaud`

Dieser Schlüssel gibt die andere Zielgruppe (`aud`) an, für die dieses ID-Token bestimmt ist. Es muss eine der OAuth 2.0-Client-IDs Ihrer Anwendung sein.

sub

Funktioniert mit Zeichenfolgenoperatoren.

Beispiele:

- `accounts.google.com:sub`
- token.actions.githubusercontent.com:sub

Verwenden Sie diese Schlüssel, um zu überprüfen, ob der Betreff mit dem Betreff übereinstimmt, den Sie in der Richtlinie angegeben haben. Sie können den `sub`-Schlüssel mit dem `aud`-Schlüssel für denselben Identitätsanbieter verwenden.

In der folgenden Rollenvertrauensrichtlinie beschränkt der `sub` Bedingungsschlüssel die Rolle auf den GitHib angegebenen Zweig`demo`.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "Condition": {
         "StringEquals": {
            "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
            "token.actions.githubusercontent.com:sub": "repo:octo-org/octo-
repo:ref:refs/heads/demo"
         }
      }
   ]
}
```

Amazon Cognito

Auf dieser Registerkarte wird erklärt, wie Amazon Cognito OIDC-Ansprüche den AWS STS Bedingungskontextschlüsseln zuordnet. AWS Mit diesen Schlüsseln können Sie den Zugriff auf eine Rolle steuern. Vergleichen Sie dazu die AWS STS Bedingungsschlüssel mit den Werten in der Spalte IdP JWT Claim.

Für Rollen, die von Amazon Cognito verwendet werden, werden Schlüssel definiert mit `cognito-identity.amazonaws.com`, gefolgt von der Forderung.

Weitere Informationen zur Zuordnung von Identitätspool-Ansprüchen finden Sie unter Standardanbieterzuordnungen im Amazon Cognito Developer Guide. Weitere Informationen zur Zuordnung von Benutzerpool-Ansprüchen finden Sie unter Verwenden des ID-Tokens im Amazon Cognito Developer Guide.

AWS STS Bedingungsschlüssel	IdP JWT-Anspruch	Während der Sitzung verfügbar
amr	amr	Ja
aud	aud	Ja
oaud	aud	Nein

AWS STS Bedingungsschlüssel	IdP JWT-Anspruch	Während der Sitzung verfügbar
sub	sub	Ja

amr

Funktioniert mit Zeichenfolgenoperatoren. Der Schlüssel enthält mehrere Werte, d. h. Sie überprüfen ihn in einer Richtlinie mithilfe von Bedingungsmengenoperatoren.

Beispiel — `cognito-identity.amazonaws.com:amr`

Die Referenz zu Authentifizierungsmethoden enthält Anmeldeinformationen über den Benutzer. Der Schlüssel kann die folgenden Werte enthalten:

- Wenn der Benutzer nicht authentifiziert wurde, enthält der Schlüssel nur `unauthenticated`.
- Wenn der Benutzer authentifiziert ist, enthält der Schlüssel den Wert `authenticated` und den Namen des Anmeldeanbieters, der im Call (`cognito-identity.amazonaws.com`) verwendet wurde.

Beispielsweise testet die folgende Bedingung in der Vertrauensrichtlinie für eine Amazon Cognito Cognito-Rolle, ob der Benutzer nicht authentifiziert ist.

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `cognito-identity.amazonaws.com:aud`

Der Benutzerpool-App-Client, der Ihren Benutzer authentifiziert hat. Amazon Cognito gibt den gleichen Wert im Zugriffstoken-Anspruch `client_id` wieder.

oaud

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `cognito-identity.amazonaws.com:oaud`

Der Benutzerpool-App-Client, der Ihren Benutzer authentifiziert hat. Amazon Cognito gibt den gleichen Wert im Zugriffstoken-Anspruch `client_id` wieder.

sub

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `cognito-identity.amazonaws.com:sub`

Eine eindeutige ID (UUID) oder ein Betreff für den authentifizierten Benutzer. Möglicherweise ist der Benutzername in Ihrem Benutzerpool nicht eindeutig. Der Unteranspruch ist der beste Weg, um einen bestimmten Benutzer zu identifizieren. Sie können den `sub`-Schlüssel mit dem `aud`-Schlüssel für denselben Identitätsanbieter verwenden.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "Condition": {
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
   ]
}
```

Login with Amazon

Auf dieser Registerkarte wird erklärt, wie Login with Amazon OIDC-Behauptungen zuordnet, um Kontextschlüssel zu AWS STS konditionieren. AWS Mit diesen Schlüsseln können Sie den Zugriff auf eine Rolle steuern. Vergleichen Sie dazu die AWS STS Bedingungsschlüssel mit den Werten in der Spalte IdP JWT Claim.

AWS STS Bedingungsschlüssel	IdP JWT-Anspruch	Während der Sitzung verfügbar
app_id	Application ID	Ja
sub	Benutzer-ID	Ja
user_id	Benutzer-ID	Ja

app_id

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `www.amazon.com:app_id`

Dieser Schlüssel gibt den Zielgruppenkontext an, der dem von anderen Identitätsanbietern verwendeten `aud` Feld entspricht.

sub

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `www.amazon.com:sub`

Dieser Schlüssel überprüft, ob die Benutzer-ID mit der übereinstimmt, die Sie in der Richtlinie angegeben haben. Sie können den `sub` -Schlüssel mit dem `aud`-Schlüssel für denselben Identitätsanbieter verwenden.

user_id

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `www.amazon.com:user_id`

Dieser Schlüssel gibt den Zielgruppenkontext an, der dem von anderen Identitätsanbietern verwendeten `aud` Feld entspricht. Sie können den `user_id` Schlüssel zusammen mit dem `id` Schlüssel für denselben Identitätsanbieter verwenden.

Facebook

Auf dieser Registerkarte wird erklärt, wie Facebook OIDC-Behauptungen zuordnet, um Kontexteingaben zu konditionieren AWS STS . AWS Mit diesen Schlüsseln können Sie den Zugriff

auf eine Rolle steuern. Vergleichen Sie dazu die AWS STS Bedingungsschlüssel mit den Werten in der Spalte IdP JWT Claim.

AWS STS Bedingungsschlüssel	IdP JWT-Anspruch	Während der Sitzung verfügbar
app_id	Application ID	Ja
id	id	Ja

app_id

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `graph.facebook.com:app_id`

Dieser Schlüssel überprüft, ob der Zielgruppenkontext mit dem `aud` Feld übereinstimmt, das von anderen Identitätsanbietern verwendet wird.

id

Funktioniert mit Zeichenfolgenoperatoren.

Beispiel — `graph.facebook.com:id`

Mit diesem Schlüssel wurde überprüft, ob die Anwendungs- (oder Site-) ID mit der ID übereinstimmt, die Sie in der Richtlinie angegeben haben.

Weitere Informationen zum OIDC-Verbund

- Amazon-Cognito-Benutzerhandbuch
- OIDC-Föderation

Verfügbare Schlüssel für SAML-basierten AWS STS -Verbund

Wenn Sie mit einem SAML-basierten Verbund mithilfe von AWS Security Token Service (AWS STS) arbeiten, können Sie zusätzliche Bedingungsschlüssel in die Richtlinie aufnehmen.

Vertrauensrichtlinien der SAML-Rolle

Sie können in der Vertrauensrichtlinie einer Rolle die folgenden Schlüssel verwenden, um festzustellen, ob der Aufrufer die Rolle übernehmen darf. Abgesehen von `saml:doc` werden alle Werte aus der SAML-Zusicherung abgeleitet. Alle Elemente in der Liste sind im visuellen Editor der IAM-Konsole verfügbar, wenn Sie eine Richtlinie mit Bedingungen erstellen oder bearbeiten. Elemente, die mit `[]` gekennzeichnet sind, können einen Wert enthalten, der eine Liste des angegebenen Typs ist.

saml:aud

Funktioniert mit Zeichenfolgenoperatoren.

Eine Endpunkt-URL, für die SAML-Zusicherungen bereitgestellt werden. Der Wert dieses Schlüssels stammt aus dem Feld `SAML Recipient` der Zusicherung, nicht aus dem Feld `Audience`.

saml:commonName[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `commonName`-Attribut.

saml:cn[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduOrg`-Attribut.

saml:doc

Funktioniert mit Zeichenfolgenoperatoren.

Dieser Schlüssel steht für den Auftraggeber, der zum Übernehmen der Rolle verwendet wurde. Das Format ist *`Account-ID`*/*`provider-friendly-name`*, z. B. `123456789012/SAMLProviderName` Der Wert Konto-ID bezieht sich auf das Konto, zu dem der SAML-Anbieter gehört.

saml:edupersonaffiliation[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonassurance[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonentitlement[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonnickname[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonorgdn

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonorgunitdn[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonprimaryaffiliation

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonprimaryorgunitdn

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonprincipalname

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersonscopedaffiliation[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:edupersontargetedid[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduPerson`-Attribut.

saml:eduorghomepageuri[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduOrg`-Attribut.

saml:eduorgidentityauthnpolicyuri[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduOrg`-Attribut.

saml:eduorglegalname[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduOrg`-Attribut.

saml:eduorgsuperioruri[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduOrg`-Attribut.

saml:eduorgwhitepagesuri[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `eduOrg`-Attribut.

saml:givenName[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `givenName`-Attribut.

saml:iss

Funktioniert mit Zeichenfolgenoperatoren.

Der Aussteller in Form eines URN

saml:mail[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `mail`-Attribut.

saml:name[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `name`-Attribut.

saml:namequalifier

Funktioniert mit Zeichenfolgenoperatoren.

Ein Hash-Wert basierend auf dem Anzeigenamen des SAML-Anbieters. Der Wert ist die Verkettung der folgenden Werte in der an gegebenen Reihenfolge und getrennt durch das Zeichen „/“:

1. Der `Issuer` Antwortwert (`saml:iss`)
2. Die `AWS`-Konto-ID.
3. Der Anzeigename (der letzte Teil des ARN) des SAML-Anbieters in IAM

Die Verkettung der Konto-ID und des Anzeigenamens des SAML-Anbieters steht als der Schlüssel `saml:doc` für IAM-Richtlinien zur Verfügung. Weitere Informationen finden Sie unter Eindeutige Identifizierung von Benutzern im SAML-basierten Verbund.

saml:organizationStatus[]

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `organizationStatus`-Attribut.

saml:primaryGroupSID[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `primaryGroupSID`-Attribut.

saml:sub

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist der Betreff des Antrags, der einen Wert enthält, der einen einzelnen Benutzer innerhalb einer Organisation eindeutig identifiziert (zum Beispiel `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Funktioniert mit Zeichenfolgenoperatoren.

Dieser Schlüssel kann den Wert `persistent` oder `transient` oder die vollständige `Format`-URI aus den in der SAML-Zusicherung verwendeten Elementen `Subject` und `NameID` enthalten. Der Wert `persistent` gibt an, dass der Wert in `saml:sub` für einen Benutzer für alle Sitzungen identisch ist. Wenn der Wert `transient` lautet, hat der Benutzer einen anderen `saml:sub`-Wert für jede Sitzung. Weitere Informationen zum `NameID`-Attribut des `Format`-Elements finden Sie unter SAML-Assertionen für die Authentifizierungsantwort konfigurieren.

saml:surname[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `surnameuid`-Attribut.

saml:uid[]

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist ein `uid`-Attribut.

saml:x500 [] UniqueIdentifier

Funktioniert mit Zeichenfolgenoperatoren.

Hierbei handelt es sich um ein `x500UniqueIdentifier`-Attribut.

Allgemeine Informationen zu den Attributen `eduPerson` und `eduOrg` finden Sie auf der Wiki-Website von REFEDS. Eine Liste der `eduPerson`-Attribute finden Sie unter eduPerson Object Class Specification (201602).

Bedingungsschlüssel mit dem Typ "Liste" können mehrere Werte enthalten. Um in einer Richtlinie Bedingungen für Listenwerte zu erstellen, können Sie Mengenoperatoren (`ForAllValues`,

`ForAnyValue`) verwenden. Um beispielsweise jeden Benutzer zuzulassen, dessen Zugehörigkeit „Lehrkörper“ oder „Personal“ (aber nicht „Student“) ist, können Sie eine Bedingung wie die folgende verwenden:

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Dienstübergreifende SAML-basierte Verbundkontextschlüssel AWS STS

Einige SAML-basierte Verbundbedingungsschlüssel können in nachfolgenden Anfragen verwendet werden, um AWS Operationen in anderen Diensten und Aufrufen zu autorisieren. `AssumeRole` Dies sind die folgenden Bedingungsschlüssel, die in Rollenvertrauensrichtlinien verwendet werden können, wenn föderierte Prinzipale eine andere Rolle übernehmen, und in Ressourcenrichtlinien anderer AWS Dienste, um den Ressourcenzugriff von Verbundprinzipalen zu autorisieren. Weitere Informationen über die Verwendung dieser Schlüssel finden Sie unter Informationen zum SAML-2.0-basierten Verbund.

Wählen Sie einen Bedingungsschlüssel aus, um die Beschreibung zu sehen.

- saml:namequalifier
- saml:sub
- saml:sub_type

> **Note**
>
> Nach der ersten Authentifizierungsantwort des externen Identitätsanbieters (IDP) stehen keine anderen SAML-basierten Verbundbedingungsschlüssel zur Verfügung.

Verfügbare Schlüssel für AWS STS

Sie können die folgenden Bedingungsschlüssel in IAM-Rollenvertrauensrichtlinien für Rollen verwenden, von denen angenommen wird, dass sie AWS Security Token Service (AWS STS) -Operationen verwenden.

saml:sub

Funktioniert mit Zeichenfolgenoperatoren.

Dies ist der Betreff des Antrags, der einen Wert enthält, der einen einzelnen Benutzer innerhalb einer Organisation eindeutig identifiziert (zum Beispiel `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

sts: AWSServiceName

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um einen Service anzugeben, in dem ein Bearertoken verwendet werden kann. Wenn Sie diesen Bedingungsschlüssel in einer Richtlinie verwenden, geben Sie den Service über einen Dienstauftraggeber an. Ein ServiceAuftraggeber ist der Name eines Services, der im Element `Principal` einer Richtlinie angegeben werden kann. Zum Beispiel `codeartifact.amazonaws.com` ist der AWS CodeArtifact Service Principal.

Für einige AWS Dienste benötigen Sie die Erlaubnis, ein AWS STS Dienstträgertoken abzurufen, bevor Sie programmgesteuert auf ihre Ressourcen zugreifen können. AWS CodeArtifact erfordert beispielsweise, dass Prinzipale Bearer-Tokens verwenden, um einige Vorgänge auszuführen. Der Befehl `aws codeartifact get-authorization-token` gibt ein Bearertoken zurück. Anschließend können Sie das Bearer-Token verwenden, um Operationen auszuführen. AWS CodeArtifact Weitere Hinweise zu Bearer-Tokens finden Sie unter Verwenden von Bearertoken.

Verfügbarkeit – Dieser Schlüssel ist in Anforderungen vorhanden, die ein Bearer-Token erhalten. Sie können nicht direkt anrufen, AWS STS um ein Bearer-Token zu erhalten. Wenn Sie einige Operationen in anderen Services ausführen, fordert der Service das Bearer-Token in Ihrem Namen an.

Sie können diesen Bedingungsschlüssel verwenden, um es Auftraggeber zu ermöglichen, ein Bearer-Token für die Verwendung mit einem bestimmten Service zu erhalten.

sts: DurationSeconds

Funktioniert mit nummerischen Operatoren.

Verwenden Sie diesen Schlüssel, um die Dauer (in Sekunden) anzugeben, die ein Principal beim Abrufen eines AWS STS Bearer-Tokens verwenden kann.

Für einige AWS Dienste benötigen Sie die Erlaubnis, ein AWS STS Dienstträgertoken abzurufen, bevor Sie programmgesteuert auf ihre Ressourcen zugreifen können. AWS CodeArtifact erfordert

beispielsweise, dass Prinzipale Bearer-Tokens verwenden, um einige Vorgänge auszuführen. Der Befehl `aws codeartifact get-authorization-token` gibt ein Bearertoken zurück. Anschließend können Sie das Bearer-Token verwenden, um Operationen auszuführen. AWS CodeArtifact Weitere Hinweise zu Bearer-Tokens finden Sie unter Verwenden von Bearertoken.

Verfügbarkeit – Dieser Schlüssel ist in Anforderungen vorhanden, die ein Bearer-Token erhalten. Sie können nicht direkt anrufen, AWS STS um ein Bearer-Token zu erhalten. Wenn Sie einige Operationen in anderen Services ausführen, fordert der Service das Bearer-Token in Ihrem Namen an. Der Schlüssel ist für AWS STS -assume-role-Operationen nicht vorhanden.

sts: ExternalId

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um zu verlangen, dass ein Auftraggeber bei der Übernahme einer IAM-Rolle einen bestimmten Bezeichner bereitstellt.

Verfügbarkeit — Dieser Schlüssel ist in der Anfrage enthalten, wenn der Principal eine externe ID bereitstellt und gleichzeitig mithilfe der AWS CLI AWS OR-API eine Rolle übernimmt.

Eine eindeutige Kennung, die erforderlich sein kann, wenn Sie eine Rolle in einem anderen Konto übernehmen. Wenn Ihnen der Administrator des Kontos, zu dem die Rolle gehört, eine externe ID zur Verfügung gestellt hat, geben Sie diesen Wert im Parameter `ExternalId` an. Dieser Wert kann eine beliebige Zeichenfolge sein, wie eine Passphrase oder Kontonummer. Die Hauptfunktion des externen Ausweises besteht darin, das Problem des verwirrten Stellvertreters zu lösen und zu verhindern. Weitere Informationen über die externe ID und den "verwirrten Stellvertreter" finden Sie unter So verwenden Sie eine externe ID, wenn Sie Dritten Zugriff auf Ihre AWS Ressourcen gewähren.

Der Wert `ExternalId` muss mindestens 2 Zeichen und darf höchstens 1.224 Zeichen lang sein. Der Wert muss alphanumerisch sein ohne Leerzeichen. Er kann auch die folgenden Zeichen enthalten: Pluszeichen (+), Gleichheitszeichen (=), Komma (,), Punkt (.), At-Zeichen (@), Doppelpunkt (:), Schrägstrich (/) und Bindestrich (-).

sts:RequestContext/Kontextschlüssel

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um die Schlüssel-Wert-Paare aus dem Sitzungskontext, die in die signierte Kontext-Assertion des vertrauenswürdigen Token-Emittenten eingebettet sind, die in der Anfrage übergeben wurde, mit den in der Rollen-Vertrauensrichtlinie angegebenen Kontext-Schlüsselwerten zu vergleichen.

Verfügbarkeit — Dieser Schlüssel ist in der Anfrage enthalten, wenn eine Kontext-Assertion im `ProvidedContexts` Anforderungsparameter angegeben wird, während eine Rolle mithilfe der AWS STS `AssumeRole` API-Operation übernommen wird.

Dieser Kontextschlüssel ist formatiert als `"sts:RequestContext/context-key":"context-value"`, wobei `context-key` und `context-value` ein Kontext-Schlüssel-Wert-Paar sind. Wenn mehrere Kontextschlüssel in die in der Anforderung übergebene signierte Kontext-Assertion eingebettet sind, gibt es einen Kontextschlüssel für jedes Schlüssel-Wert-Paar. Sie müssen in der Rollen-Vertrauensrichtlinie die Erlaubnis für die `sts:SetContext`-Aktion erteilen, damit ein Prinzipal Kontextschlüssel innerhalb des resultierenden Sitzungs-Token festlegen kann. Weitere Informationen zu den unterstützten IAM Identity Center-Kontextschlüsseln, die mit diesem Schlüssel verwendet werden können, finden Sie im Benutzerhandbuch unter AWS STS Bedingungsschlüssel für IAM Identity Center.AWS IAM Identity Center

Sie können diesen Schlüssel in einer Rollen-Vertrauensrichtlinie verwenden, um eine detaillierte Zugriffskontrolle zu erzwingen, die auf dem Benutzer oder seinen Attributen basiert, wenn er eine Rolle übernimmt. Sie können Amazon Redshift beispielsweise als IAM-Identity-Center-Anwendung konfigurieren, um im Namen Ihrer Belegschaft oder Verbunds-Identitäten auf Amazon-S3-Ressourcen zuzugreifen.

Die folgende Rollen-Vertrauensrichtlinie erlaubt es dem Amazon-Redshift-Service-Prinzipal, eine Rolle im Konto `111122223333` zu übernehmen. Sie erteilt dem Amazon-Redshift-Service-Prinzipal auch die Berechtigung, Kontextschlüssel in der Anfrage festzulegen, sofern der Wert für den Kontextschlüssel `identitystore:UserId` auf `1111-22-3333-44-5555` gesetzt ist. Nachdem die Rolle übernommen wurde, erscheint die Aktivität in den AWS CloudTrail Protokollen innerhalb des `AdditionalEventData` Elements, die die Schlüssel-Wert-Paare für den Sitzungskontext enthalten, die vom Kontextanbieter in der Anfrage „Rolle annehmen“ festgelegt wurden. Dies erleichtert Administratoren die Unterscheidung zwischen Rollensitzungen, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird. Die Schlüssel-Wert-Paare werden vom angegebenen Kontextanbieter festgelegt, nicht von oder. AWS CloudTrail AWS STS Dadurch hat der Kontextanbieter die Kontrolle darüber, welcher Kontext in den CloudTrail Protokollen und Sitzungsinformationen enthalten ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "redshift.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:SetContext"
            ],
            "Condition": {
                "ForAllValues:ArnEquals": {
                    "sts:RequestContextProviders": [
                        "arn:aws:iam::aws:contextProvider/IdentityCenter"
                    ]
                },
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "sts:RequestContext/identitystore:UserId":
 "1111-22-3333-44-5555"
                }
            }
        }
    ]
}
```

sts: RequestContextProviders

Funktioniert mit ARN-Operatoren.

Verwenden Sie diesen Schlüssel, um den Kontextanbieter-ARN in der Anfrage mit dem Kontextanbieter-ARN zu vergleichen, der in der Rollen-Vertrauensrichtlinie angegeben ist.

Verfügbarkeit — Dieser Schlüssel ist in der Anfrage vorhanden, wenn eine Kontext-Assertion im `ProvidedContexts` Anforderungsparameter angegeben wird, während eine Rolle mithilfe der AWS STS `AssumeRole` API-Operation übernommen wird.

Die folgende Beispielbedingung überprüft, ob der in der Anforderung übergebene Kontextanbieter-ARN mit dem in der Rollen-Vertrauensrichtlinien-Bedingung angegebenen ARN übereinstimmt.

```
      "Condition": {
        "ForAllValues:ArnEquals": {
          "sts:RequestContextProviders": [
            "arn:aws:iam::aws:contextProvider/IdentityCenter"
          ]
```

```
        }
      }
```

sts: RoleSessionName

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Sitzungsnamen zu vergleichen, den ein Auftraggeber angibt, wenn er eine Rolle mit dem Wert annimmt, der in der Richtlinie angegeben ist.

Verfügbarkeit — Dieser Schlüssel ist in der Anfrage enthalten, wenn der Principal die Rolle mithilfe eines beliebigen Assume-Role-CLI-Befehls oder einer AWS STS `AssumeRole` API-Operation übernimmt. AWS Management Console

Sie können diesen Schlüssel in einer Rollenvertrauensrichtlinie verwenden, um zu verlangen, dass Ihre Benutzer einen bestimmten Sitzungsnamen angeben, wenn sie eine Rolle übernehmen. Sie können beispielsweise verlangen, dass IAM-Benutzer ihren eigenen Benutzernamen als Sitzungsnamen angeben. Nachdem der IAM-Benutzer die Rolle übernommen hat, wird die Aktivität in AWS CloudTrail -Protokollen mit dem Sitzungsnamen angezeigt, der dem Benutzernamen entspricht. Dies erleichtert Administratoren die Unterscheidung zwischen Rollensitzungen, wenn eine Rolle von verschiedenen Auftraggeber verwendet wird.

Die folgende Rollenvertrauensrichtlinie erfordert, dass IAM-Benutzer im Konto `111122223333` ihren IAM-Benutzernamen als Sitzungsnamen angeben, wenn sie die Rolle übernehmen. Diese Anforderung wird mit der Bedingungsvariablen `aws:username` im Bedingungsschlüssel erzwungen. Mit dieser Richtlinie können IAM-Benutzer die Rolle übernehmen, an die die Richtlinie angefügt ist. Diese Richtlinie erlaubt niemandem, der temporäre Anmeldeinformationen verwendet, die Rolle zu übernehmen, da die `username`-Variable nur für IAM-Benutzer vorhanden ist.

> **Important**
>
> Sie können jeden einwertigen Bedingungsschlüssel als Variable verwenden. Sie können einen mehrwertigen Bedingungsschlüssel nicht als Variable verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "${aws:username}"}
            }
        }
    ]
}
```

Wenn ein Administrator das AWS CloudTrail Protokoll einer Aktion einsieht, kann er den Sitzungsnamen mit den Benutzernamen in seinem Konto vergleichen. Im folgenden Beispiel führte der Benutzer `matjac` den Vorgang mit der Rolle namens `MateoRole`. Der Administrator kann sich dann an Mateo Jackson wenden, den der Benutzer als `matjac` benannt hat.

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

Wenn Sie kontenübergreifenden Zugriff mithilfe von Rollen zulassen, können Benutzer in einem Konto eine Rolle in einem anderen Konto übernehmen. Der ARN des Benutzers mit angenommener Rolle, der unter aufgeführt ist, CloudTrail beinhaltet das Konto, in dem die Rolle existiert. Darin ist nicht das Konto des Benutzers enthalten, der die Rolle übernommen hat. Benutzer sind nur innerhalb eines Kontos eindeutig. Daher empfehlen wir, diese Methode nur für die Überprüfung von CloudTrail Protokollen für Rollen zu verwenden, die von Benutzern in von Ihnen verwalteten Konten übernommen wurden. Ihre Benutzer verwenden möglicherweise denselben Benutzernamen in mehreren Konten.

sts: SourceIdentity

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Sitzungsnamen zu vergleichen, den ein Auftraggeber angibt, wenn er eine Rolle mit dem Wert annimmt, der in der Richtlinie angegeben ist.

Verfügbarkeit — Dieser Schlüssel ist in der Anfrage enthalten, wenn der Principal eine Quellidentität bereitstellt und gleichzeitig eine Rolle mithilfe eines AWS STS Assume-Role-CLI-Befehls oder einer AWS STS `AssumeRole` API-Operation annimmt.

Sie können diesen Schlüssel in einer Rollenvertrauensrichtlinie verwenden, um zu verlangen, dass Ihre Benutzer eine bestimmte Ursprungsidentität festlegen, wenn sie eine Rolle übernehmen. Beispielsweise können Sie verlangen, dass Ihre Mitarbeiter oder Verbundidentitäten einen Wert für die Quellidentität angeben. Sie können Ihren Identity Provider (IdP) so konfigurieren, dass eines der Attribute verwendet wird, die Ihren Benutzern zugeordnet sind, z. B. ein Benutzername oder eine E-Mail als Quellidentität. Der IdP übergibt dann die Quellidentität als Attribut in den Assertions oder Claims, an die er sendet. AWS Der Wert des Quellidentitätsattributs identifiziert den Benutzer oder die Anwendung, der die Rolle übernimmt.

Nachdem der Benutzer die Rolle übernommen hat, erscheint die Aktivität in den AWS CloudTrail -Protokollen mit dem eingestellten Wert der Quellidentität. Auf diese Weise können Administratoren leichter feststellen, wer oder was Aktionen mit einer Rolle in ausgeführt hat. AWS Sie müssen für die Aktion `sts:SetSourceIdentity` die Berechtigung erteilen, dass eine Identität eine Quellidentität festlegen kann.

Im Gegensatz zu `sts:RoleSessionName`, nachdem die Quellidentität festgelegt wurde, kann der Wert nicht geändert werden. Es ist im Anforderungskontext für alle Aktionen vorhanden, die mit der Rolle von der Quellidentität durchgeführt werden. Wenn Sie die Sitzungsanmeldeinformationen verwenden, bleibt der Wert in nachfolgenden Rollensitzungen bestehen, um eine andere Rolle anzunehmen. Die Annahme einer Rolle von einer anderen wird als Rollenverkettung bezeichnet.

Sie können den `aws:SourceIdentity`globalen Bedingungsschlüssel verwenden, um den Zugriff auf AWS Ressourcen anhand des Werts der Quellidentität in nachfolgenden Anfragen weiter zu steuern.

Die folgende Rollenvertrauensrichtlinie erlaubt dem IAM-Benutzer `AdminUser`, eine Rolle im Konto `111122223333` zu übernehmen. Sie erteilt dem `AdminUser` auch die Erlaubnis, eine Quellidentität zu setzen, solange die gesetzte Quellidentität `DiegoRamirez` ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
```

```
            ],
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

Weitere Informationen zur Verwendung von Quellidentitätsinformationen finden Sie unter Überwachen und Steuern von Aktionen mit übernommenen Rollen.

sts: TransitiveTagKeys

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um die transitiven Sitzungstag-Schlüssel in der Anforderung mit den in der Richtlinie angegebenen Schlüsseln zu vergleichen.

Verfügbarkeit – Dieser Schlüssel ist in der Anforderung vorhanden, wenn Sie eine Anforderung mit temporären Sicherheitsanmeldeinformationen stellen. Dazu gehören Anmeldeinformationen, die mit einer beliebigen assume-role-Operation oder der `GetFederationToken`-Operation erstellt wurden.

Wenn Sie eine Anforderung mit temporären Sicherheitsanmeldeinformationen erstellen, enthält der Anforderungskontext den `aws:PrincipalTag`-Kontextschlüssel. Dieser Schlüssel enthält eine Liste von Sitzungs-Tags, transitiven Sitzungs-Tags und Rollentags. Transitive Sitzungs-Tags sind Tags, die in allen nachfolgenden Sitzungen bestehen, wenn Sie die Sitzungsanmeldeinformationen verwenden, um eine andere Rolle anzunehmen. Die Annahme einer Rolle von einer anderen wird als Rollenverkettung bezeichnet.

Sie können diesen Bedingungsschlüssel in einer Richtlinie verwenden, um bestimmte Sitzungs-Tags als transitiv festzulegen, wenn Sie eine Rolle übernehmen oder einen Benutzer föderieren.

Aktionen, Ressourcen und Bedingungsschlüssel für AWS Dienste

Jeder AWS Dienst kann Aktionen, Ressourcen und Bedingungskontextschlüssel für die Verwendung in IAM-Richtlinien definieren. Eine Liste der AWS Dienste und ihrer Aktionen, Ressourcen und Bedingungskontextschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel in der Service Authorization Reference.

Nutzen Sie Ressourcen, um mehr über IAM zu erfahren

IAM ist ein umfangreiches Produkt, und Sie werden viele Ressourcen finden, die Ihnen helfen, mehr darüber zu erfahren, wie IAM Ihnen helfen kann, Ihre AWS-Konto Ressourcen zu schützen.

Themen

Identitäten

Lesen Sie diese Ressourcen zum Erstellen, Verwalten und Verwenden von Identitäten.

- Verwalten von Identitäten in IAM Identity Center – Verfahrensinformationen zum Erstellen von Benutzern und Gruppen in IAM Identity Center.
- IAM-Identitäten (Benutzer, Gruppen und Rollen) – Eine umfassende Beschreibung über Benutzer, Gruppen und Rollen.

Anmeldeinformationen (Passwörter, Zugriffsschlüssel und MFA-Geräte)

Lesen Sie die folgenden Anleitungen zur Verwaltung von Passwörtern, Zugriffsschlüsseln und MFA-Geräten für Sie AWS-Konto und für IAM-Benutzer.

- Benutzerpasswörter verwalten in AWS – Hier werden die Optionen zur Verwaltung von Passwörtern für IAM-Benutzer in Ihrem Konto beschrieben.
- Verwalten der Zugriffsschlüssel für IAM-Benutzer – Beschreibt, wie Zugriffsschlüssel funktionieren und wie Sie sie verwenden können, um programmatische Aufrufe von AWS zu tätigen. Es gibt andere sicherere Alternativen als Zugangsschlüssel, die Sie zunächst in Betracht ziehen sollten.

Weitere Informationen finden Sie unter Überlegungen und Alternativen für Schlüssel für den langfristigen Zugriff im Allgemeine AWS-Referenz -Handbuch.

- Verwendung der Multi-Faktor-Authentifizierung (MFA) in AWS – In diesem Thema wird beschrieben, wie Ihr Konto und Ihre IAM-Benutzer so konfiguriert werden, dass sowohl ein Passwort als auch ein einmaliger Code erforderlich sind, der auf einem Gerät generiert wird, bevor die Anmeldung zulässig ist. (Dies wird gelegentlich auch als Zwei-Faktor-Authentifizierung bezeichnet.)

Allgemeine Informationen zu den Arten von Anmeldeinformationen, die Sie für den Zugriff auf Amazon Web Services verwenden, finden Sie unter AWS -Sicherheitsanmeldeinformationen im Allgemeine AWS-Referenz -Handbuch.

Berechtigungen und Richtlinien

Hier lernen Sie die innere Funktionsweise von IAM-Richtlinien kennen und erhalten Tipps zu den besten Methoden für die Erteilung von Berechtigungen:

- Berechtigungen und Richtlinien in IAMDieses Thema bietet eine Einführung in die Richtliniensprache, die zum Definieren von Berechtigungen verwendet wird. Darin wird beschrieben, wie Berechtigungen an Benutzer oder Gruppen bzw. bei einigen AWS -Produkten an Ressourcen selbst angefügt werden können.
- IAM-JSON-RichtlinienelementreferenzDieses Thema enthält Beschreibungen und Beispiele für jedes Richtliniensprachenelement.
- Validieren von IAM-Richtlinien – Suchen von Ressourcen für die JSON-Richtlinienvalidierung.
- Beispiele für identitätsbasierte Richtlinien in IAM— Zeigt Beispiele für Richtlinien für allgemeine Aufgaben in verschiedenen AWS Produkten.
- AWS Policy Generator – Hiermit können Sie benutzerdefinierte Richtlinien erstellen, indem Sie Produkte und Aktionen aus einer Liste auswählen.
- IAM Policy Simulator — Testen Sie, ob eine Richtlinie eine bestimmte Anfrage an AWS zulässt oder ablehnt.

Verbund und Delegierung

Sie können Benutzern, die an anderer Stelle authentifiziert (angemeldet) sind, Zugriff auf Ihre AWS-Konto Ressourcen gewähren. Dies können IAM-Benutzer in einem anderen Bereich sein AWS-Konto

(bekannt als Delegierung), Benutzer, die mit dem Anmeldeprozess Ihrer Organisation authentifiziert wurden, oder Benutzer von einem Internet-Identitätsanbieter wie Login with Amazon, Facebook, Google oder einem anderen OpenID Connect (OIDC) -kompatiblen Identitätsanbieter. In diesen Fällen erhalten die Benutzer temporäre Sicherheitsanmeldedaten für den Zugriff auf Ressourcen. AWS

- Tutorial: Delegieren des Zugriffs in allen AWS -Konten mithilfe von IAM-Rollen – Dieses Tutorial führt Sie durch die Schritte, wie Sie einem IAM-Benutzer in einem anderen AWS-Konto kontoübergreifenden Zugriff gewähren.
- Gängige Szenarien für temporäre Anmeldeinformationen— Beschreibt Methoden, mit denen Benutzer verbunden werden können, AWS nachdem sie sich außerhalb von authentifiziert haben. AWS

IAM und andere Produkte AWS

Die meisten AWS Produkte sind in IAM integriert, sodass Sie IAM-Funktionen verwenden können, um den Zugriff auf die Ressourcen in diesen Produkten zu schützen. In den folgenden Ressourcen werden IAM und Sicherheit für einige der beliebtesten Produkte behandelt. AWS Eine vollständige Liste der Produkte, die mit IAM verwendet werden können, sowie Links zu weiteren Informationen über beide Services finden Sie unter AWS Dienste, die mit IAM funktionieren.

Verwenden von IAM mit Amazon EC2

- Controlling Access to Amazon EC2 Resources – In diesem Thema wird die Verwendung der IAM-Funktionen beschrieben, mit denen Benutzer Amazon EC2–Instances, -Volumes und vieles mehr verwalten können.
- Verwenden von Instance-Profilen— Beschreibt, wie IAM-Rollen verwendet werden, um Anmeldeinformationen für Anwendungen, die auf Amazon EC2 EC2-Instances ausgeführt werden und Zugriff auf andere AWS Produkte benötigen, sicher bereitzustellen.

Verwenden von IAM mit Amazon S3

- Verwalten von Zugriffsberechtigungen für Ihre IAM-Ressourcen – In diesem Thema wird das Amazon S3-Sicherheitsmodell für Buckets und Objekte erläutert, in denen IAM-Richtlinien enthalten sind.

- Writing IAM Policies: Grant Access to User-Specific Folders in an Amazon S3 Bucket – In diesem Blog wird erläutert, wie Benutzer ihre eigenen Ordner in Amazon S3 schützen können. (Um weitere Beiträge zu Amazon S3 und IAM aufzurufen, klicken Sie auf den Tag S3 unter dem Titel des Blogbeitrags.)

Verwenden von IAM mit Amazon RDS

- Verwenden von AWS Identity and Access Management (IAM) zur Verwaltung des Zugriffs auf Amazon RDS-Ressourcen — Beschreibt, wie Sie mit IAM den Zugriff auf Datenbank-Instances, Datenbank-Snapshots und mehr steuern können.
- A Primer on RDS Resource-Level Permissions – In diesem Blogbeitrag wird die Verwendung von IAM zum Steuern des Zugriffs auf bestimmte Amazon RDS-Instances erörtert.

Verwenden von IAM mit Amazon DynamoDB

- Verwenden von zum Steuern des Zugriffs auf IAM-Ressourcen – In diesem Thema wird beschrieben, wie Sie verwenden können, um Benutzern die Verwaltung von DynamoDB-Tabellen und -Indizes zu ermöglichen.
- Im folgenden Video (8:55) wird die Einrichtung der Zugriffssteuerung für einzelne DynamoDB-Datenbankelemente oder -attribute (oder beide) demonstriert.

  Erste Schritte mit fein abgestufter Zugriffskontrolle für DynamoDB

Allgemeine Sicherheitsverfahren

Hier finden Sie Expertentipps und Anleitungen zu den besten Möglichkeiten, Ihre Ressourcen und Ihre AWS-Konto Ressourcen zu schützen:

- Bewährte Methoden für Sicherheit, Identität und Compliance — Hier finden Sie Ressourcen zur Verwaltung der Sicherheit AWS-Konten für alle Produkte, darunter Vorschläge zur Sicherheitsarchitektur, zur Verwendung von IAM, Verschlüsselung und Datensicherheit und mehr.
- Identity and Access Management — Das AWS Well-Architected Framework hilft Ihnen dabei, die wichtigsten Konzepte, Entwurfsprinzipien und architektonischen Best Practices für das Entwerfen und Ausführen von Workloads in der Cloud zu verstehen.

- Bewährte Methoden für die Sicherheit in IAM – Hier finden Sie Empfehlungen zu den Verwendungsmöglichkeiten von IAM zum Schützen Ihrer AWS-Konto und Ressourcen.
- AWS CloudTrail Benutzerhandbuch — Wird verwendet AWS CloudTrail , um den Verlauf von API-Aufrufen nachzuverfolgen AWS und diese Informationen in Protokolldateien zu speichern. Auf diese Weise können Sie einfacher bestimmen, welche Benutzer und Konten auf Ressourcen in Ihrem Konto zugegriffen haben, wann die Aufrufe erfolgten, welche Aktionen angefordert wurden, und vieles mehr.

Allgemeine -Ressourcen

Erkunden Sie die folgenden Ressourcen, um mehr über IAM und AWS zu erfahren.

- Produktinformationen zu IAM – Allgemeine Informationen zum AWS Identity and Access Management -Produkt.
- AWS re:Post für AWS Identity and Access Management — Besuchen Sie uns AWS re:Post , um technische Fragen zu IAM mit der AWS Community zu besprechen.

- 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 Fragen zu 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.

Aufrufen der IAM-API mithilfe von HTTP-Abfrageanforderungen

Inhalt

Sie können mithilfe der Query-API programmgesteuert auf das IAM und die AWS STS Dienste zugreifen. Abfrage-API-Anforderungen sind HTTPS-Anforderungen, in denen eine auszuführende Aktion mittels eines `Action`-Parameters angegeben wird. IAM und AWS STS unterstützt GET- und POST-Anfragen für alle Aktionen. Das heißt, die API verlangt nicht, dass Sie für einige Aktionen GET und für andere POST verwenden. GET-Anforderungen unterliegen jedoch den Längenbeschränkungen für URLs. Diese Beschränkung ist abhängig vom verwendeten Browser, beträgt in der Regel jedoch 2.048 Bytes. Für größere Abfrage-API-Anforderungen muss daher eine POST-Anforderung verwendet werden.

Die Antwort erfolgt in Form eines XML-Dokuments. Weitere Informationen über die Antwort finden Sie auf den Seiten zu den einzelnen Aktionen in der IAM API-Referenz oder die AWS Security Token Service -API-Referenz.

> **Tip**
>
> Anstatt die IAM- oder AWS STS API-Operationen direkt aufzurufen, können Sie eines der AWS SDKs verwenden. Die AWS SDKs bestehen aus Bibliotheken und Beispielcode für verschiedene Programmiersprachen und Plattformen (Java, Ruby, .NET, iOS, Android usw.). Die SDKs bieten eine bequeme Möglichkeit, programmatischen Zugriff auf IAM und zu erstellen. AWS Mithilfe der SDKs lassen sich unter anderem Anforderungen kryptografisch signieren (siehe unten), Fehler verwalten und Anforderungen automatisch wiederholen. Informationen zu den AWS SDKs, einschließlich deren Download und Installation, finden Sie auf der Seite Tools für Amazon Web Services.

Weitere Informationen über die API-Vorgänge und Fehler finden Sie in der IAM API-Referenz oder die AWS Security Token Service -API-Referenz.

Endpunkte

IAM und AWS STS beide haben einen einzigen globalen Endpunkt:

- (IAM)https://iam.amazonaws.com
- (AWS STS)https://sts.amazonaws.com

> **Note**
>
> AWS STS unterstützt zusätzlich zum globalen Endpunkt auch das Senden von Anfragen an regionale Endpunkte. Bevor Sie AWS STS in einer Region verwenden können, müssen Sie STS zunächst in dieser Region für Ihre AWS-Konto aktivieren. Weitere Informationen zur Aktivierung zusätzlicher Regionen für AWS STS finden Sie unterVerwaltung AWS STS in einem AWS-Region.

Weitere Informationen zu AWS Endpunkten und Regionen für alle Dienste finden Sie unter Dienstendpunkte und Kontingente in der. Allgemeine AWS-Referenz

HTTPS erforderlich

Die Abfrage-API gibt vertrauliche Informationen wie Sicherheitsanmeldeinformationen zurück; daher müssen für alle API-Anforderungen HTTPS verwenden.

Signieren von IAM-API-Anforderungen

Anforderungen müssen über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel signiert werden. Wir raten nachdrücklich davon ab, die Root-Benutzer des AWS-Kontos -Anmeldeinformationen für die tägliche Arbeit mit IAM zu verwenden. Sie können die Anmeldeinformationen für einen IAM-Benutzer verwenden AWS STS oder temporäre Sicherheitsanmeldeinformationen generieren.

Um Ihre API-Anfragen zu signieren, empfehlen wir die Verwendung von AWS Signature Version 4. Weitere Informationen zu Signaturversion 4 finden Sie unter Signaturprozess mit Signaturversion 4 in der Allgemeinen Referenz zu AWS .

Wenn Sie Signaturversion 2 verwenden müssen, finden Sie dazu ebenfalls Informationen in der Allgemeinen Referenz zu AWS.

Weitere Informationen finden Sie hier:

- AWS Sicherheitsnachweise. Enthält allgemeine Informationen zu den Arten von Anmeldeinformationen, die für den Zugriff verwendet werden AWS.
- Bewährte Methoden für die Sicherheit in IAM. Enthält eine Liste mit Vorschlägen zur Verwendung des IAM-Dienstes zur Sicherung Ihrer AWS Ressourcen.
- Temporäre IAM Sicherheitsanmeldeinformationen. Beschreibt die Erstellung und Verwendung von temporären Sicherheitsanmeldeinformationen.

Dokumentverlauf für IAM

Die folgende Tabelle beschreibt die wesentlichen Dokumentationsupdates für IAM.

Änderung	Beschreibung	Datum
`AccessAnalyzerServiceRolePolicy` – hat neue Berechtigung	IAM Access Analyzer hat die Unterstützung für Berechtigungen zum Abrufen von Informationen über IAM-Benutzer- und Rollenrichtlinien zu den Berechtigungen auf Dienstebene von Policy hinzugefügt. AccessAnalyzerServiceRole	30. Mai 2024
`AccessAnalyzerServiceRolePolicy` – hat neue Berechtigung	IAM Access Analyzer hat die Unterstützung für Berechtigungen zum Abrufen des aktuellen Status des Block Public Access für Amazon EC2-Snapshots zu den Service-Level-Berechtigungen von Policy hinzugefügt. AccessAnalyzerServiceRole	23. Januar 2024
`AccessAnalyzerServiceRolePolicy` – hat neue Berechtigung	IAM Access Analyzer hat DynamoDB-Streams und -Tabellen zu den Dienstebenenberechtigungen von Policy hinzugefügt. AccessAnalyzerServiceRole	11. Januar 2024
`AccessAnalyzerServiceRolePolicy` – hat neue Berechtigung	IAM Access Analyzer hat Amazon S3 S3-Verzeichnis-Buckets zu den Service-Level-Berechtigungen von Policy	1. Dezember 2023

	hinzugefügt. AccessAnalyzer ServiceRole	
IAMAccessAnalyzerReadOnlyAccess – hat neue Berechtigung	IAM Access Analyzer hat IAM AccessAnalyzer ReadOnly Access Berechtigungen hinzugefügt, mit denen Sie überprüfen können, ob Aktualisierungen Ihrer Richtlinien zusätzlichen Zugriff gewähren. Diese Berechtigung wird von IAM Access Analyzer benötigt, um Richtlinienprüfungen für Ihre Richtlinien durchzuführen.	26. November 2023
IAM Access Analyzer bietet jetzt Analysatoren für ungenutzten Zugriff	IAM Access Analyzer vereinfacht die Überprüfung von ungenutztem Zugriff und führt Sie so zur geringsten Berechtigung. IAM Access Analyzer analysiert kontinuierlich Ihre Konten, um ungenutzten Zugriff zu erkennen, und erstellt ein zentrales Dashboard mit den Ergebnissen.	26. November 2023
IAM Access Analyzer bietet jetzt benutzerdefinierte Richtlinienüberprüfungen	IAM Access Analyzer bietet jetzt benutzerdefinierte Richtlinienprüfungen, um vor der Bereitstellung zu überprüfen, ob IAM-Richtlinien Ihren Sicherheitsstandards entsprechen.	26. November 2023

AccessAnalyzerServiceRolePolicy – hat neue Berechtigung	IAM Access Analyzer hat den Berechtigungen auf Dienstebene von AccessAnalyzerServiceRolePolicy IAM-Aktionen hinzugefügt, um die folgenden Aktionen zu unterstützen: • Auflisten von Entitäten für eine Richtlinie • Generieren von Informationen zum letzten Zugriff auf Services • Auflisten der wichtigsten Informationen über den Zugriffsschlüssel	26. November 2023
Informationen zur zuletzt aufgerufenen Aktion und Unterstützung bei der Richtliniengenerierung für über 60 zusätzliche Services und Aktionen	IAM unterstützt jetzt Informationen zur zuletzt aufgerufenen Aktion und generiert Richtlinien mit Informationen auf Aktionsebene für über 60 zusätzliche Services sowie eine Liste der Aktionen, für die Informationen zur zuletzt aufgerufenen Aktion verfügbar sind.	1. November 2023
Unterstützung für Informationen zur zuletzt aufgerufenen Aktion für über 140 Services	IAM bietet jetzt Informationen zur zuletzt aufgerufenen Aktion für mehr als 140 Services sowie eine Liste der Aktionen, für die Informationen zur zuletzt aufgerufenen Aktion verfügbar sind.	14. September 2023

Support für Multi-Faktor-Authentifizierung (MFA)-Geräte für Root-Benutzer und IAM-Benutzer	Jetzt können Sie bis zu acht MFA-Geräte pro Benutzer hinzufügen, darunter FIDO-Sicherheitsschlüssel, zeitgesteuertes Einmalpasswort (TOTP) mit virtuellen Authentifizierungsanwendungen oder Hardware-TOTP-Token.	16. November 2022
IAM-Access-Analyzer-Support für neue Ressourcentypen	IAM Access Analyzer hat Support für die folgenden Ressourcentypen hinzugefügt: • Amazon-EBS-Volume-Snapshots • Amazon-ECR-Repositorys • Amazon-EFS-Dateisysteme • Amazon-RDS-DB-Snapshots • Snapshots des Amazon-RDS-DB-Clusters • Amazon SNS-Themen	25. Oktober 2022
U2F ist veraltet und /FIDO-Update WebAuthn	Die Erwähnung von U2F als MFA-Option wurde entfernt und Informationen zu WebAuthn FIDO2- und FIDO-Sicherheitsschlüsseln hinzugefügt.	31. Mai 2022

Aktualisierungen der Resilienz in IAM	Es wurden Informationen zur Aufrechterhaltung des Zugriffs auf IAM-Anmeldeinformationen hinzugefügt, wenn ein Ereignis die Kommunikation zwischen AWS-Regionen unterbricht.	16. Mai 2022
Neue globale Bedingungsschlüssel für Ressourcen	Sie können jetzt den Zugriff auf Ressourcen auf der Grundlage des Kontos, der Organisationseinheit (OU) oder der Organisation steuern, in der sich Ihre Ressourcen befinden. AWS Organizations Sie können die globalen Bedingungsschlüssel `aws:ResourceAccount` , `aws:ResourceOrgID` und `aws:ResourceOrgPaths` in einer IAM-Richtlinie verwenden.	27. April 2022

Codebeispiele für IAM mit SDKs AWS	Es wurden Codebeispiele hinzugefügt, die zeigen, wie IAM mit einem AWS Software Development Kit (SDK) verwendet wird. Die Beispiele sind in Codeauszüge unterteilt, die Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen können, und Beispiele, die Ihnen zeigen, wie Sie eine bestimmte Aufgabe ausführen können, indem Sie mehrere Funktionen innerhalb desselben Services aufrufen.	7. April 2022
Richtlinienaktualisierungen des Logik-Flussdiagramms	Aktualisierungen des Flussdiagramms zur Richtlinienauswertung und des verwandten Textes im Abschnitt Bestimmen, ob eine Anforderung innerhalb eines Kontos zugelassen oder abgelehnt wird erstellen.	17. November 2021

Aktualisierungen von bewährten Sicherheitsmethoden	Es wurden Informationen über die Erstellung von administrativen Benutzern anstelle der Verwendung von Root-Benutzeranmeldeinformationen hinzugefügt, die bewährte Praxis der Verwendung von Benutzergruppen für die Zuweisung von Berechtigungen an IAM-Benutzer wurde entfernt. Außerdem wurde klargestellt, wann verwaltete Richtlinien anstelle von Inline-Richtlinien verwendet werden sollten.	5. Oktober 2021
Aktualisierungen des Themas zur Richtlinienauswertung für ressourcenbasierte Richtlinien	Es wurden Informationen über die Auswirkungen ressourcenbasierter Richtlinien und verschiedener Haupttypen im selben Konto hinzugefügt.	5. Oktober 2021
Aktualisierungen von einzelwertigen und mehrwertigen Bedingungsschlüsseln	Die Unterschiede zwischen einzelwertigen und mehrwertigen Bedingungsschlüsseln werden nun ausführlicher erläutert. Der Werttyp wurde jedem AWS globalen Bedingungskontextschlüssel hinzugefügt.	30. September 2021

IAM Access Analyzer unterstützt Amazon S3 regionübergreifende Zugriffspunkte	IAM Access Analyzer identifiziert Amazon S3 Buckets, die öffentlichen und kontoübergreifenden Zugriff ermöglichen, einschließlich solcher, die Amazon S3 regionenübergreifende Zugriffspunkte verwenden.	2. September 2021
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer vorhandenen Richtlinie	IAM Access Analyzer hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	2. September 2021
Weitere Dienste werden für die Richtliniengenerierung auf Aktionsebene unterstützt	IAM Access Analyzer kann IAM-Richtlinien mit Informationen zu Zugriffsaktivitäten auf Aktionsebene für zusätzliche Dienste generieren. AWS	24. August 2021
IAM-Richtlinien für kontoübergreifende Trails	Sie können IAM Access Analyzer jetzt verwenden, um anhand Ihrer Zugriffsaktivität detaillierte Richtlinien zu generieren, indem Sie einen AWS CloudTrail Trail in einem anderen Konto verwenden, z. B. einen zentralen Trail. AWS Organizations	18. August 2021

IAM Access Analyzer-Richtlinienüberprüfungen	IAM Access Analyzer hat die Richtlinienvalidierung erweitert, indem neue Richtlinienprüfungen hinzugefügt werden, die Bedingungen in IAM-Richtlinien überprüfen. Diese Prüfungen analysieren den Bedingungsblock in Ihrer Richtlinienanweisung und melden Sicherheitswarnungen, Fehler und Vorschläge sowie umsetzbare Empfehlungen. IAM Access Analyzer hat die folgenden Richtlinienüberprüfungen hinzugefügt: • Fehler – Ungültiges Dienstauftraggeberformat • Fehler – Fehlender Tag-Schlüssel in Bedingung • Sicherheitswarnung — Ablehnen, wenn der NotAction Tag-Bedingungsschlüssel für den Service nicht unterstützt wird • Sicherheitswarnung – Verweigern mit nicht unterstütztem Tag-Bedingungsschlüssel für den Dienst • Sicherheitswarnung – Fehlende gekoppelte Zustandsschlüssel	29. Juni 2021

	• Vorschlag — Zulassen, wenn NotAction der Tag-Bedingungsschlüssel für den Service nicht unterstützt wird • Vorschlag – Zulassen mit nicht unterstütztem Tag-Bedingungsschlüssel für Service	
Aktion, auf die zuletzt zugegriffen wurde, für weitere Dienste	Sie können jetzt in der IAM-Konsole Informationen über den letzten Zugriff auf eine Aktion anzeigen, wenn ein IAM-Auftraggeber eine Aktion für die folgenden Dienste verwendet hat: Amazon EC2, IAM, Lambda und Amazon S3 Verwaltungsaktionen. Sie können auch die AWS API AWS CLI oder verwenden, um einen Datenbericht abzurufen . Mit diesen Informationen können Sie unnötige Berechtigungen identifizieren, sodass Sie die IAM-Richtlinien besser an die Regel der geringsten Rechte anpassen können.	19. April 2021

Überwachen und Steuern von Aktionen mit übernommenen Rollen	Administratoren können IAM-Rollen so konfiguri eren, dass Identitäten eine Quellidentität übergeben müssen, die in AWS CloudTrai l protokolliert wird. Durch das Überprüfen von Informationen zur Quellidentität können Administratoren feststellen, wer oder was Aktionen mit angenommenen Rollensit zungen ausgeführt hat.	13. April 2021
Generieren von IAM-Richt linien basierend auf Zugriffsa ktivitäten	Sie können jetzt den IAM Access Analyzer verwenden , um fein abgestufte Richtlini en auf der Grundlage Ihrer Zugriffsaktivitäten in Ihrer AWS CloudTrail zu erstellen.	7. April 2021
IAM Access Analyzer-Richtlini enüberprüfungen	IAM Access Analyzer bietet jetzt mehr als 100 Richtlinienüberprüfungen mit umsetzbaren Empfehlungen während der Richtlinienerstell ung.	16. März 2021
Erweiterte Optionen für die Richtlinien	Erweiterte Richtlinienvalidie rung, verfügbar in der IAM-Konsole, in der AWS API und AWS CLI mithilfe von Richtlinienprüfungen im IAM Access Analyzer, um Ihnen bei der Erstellung sicherer und funktionaler JSON-Richtlinien zu helfen.	15. März 2021

Markieren von IAM-Ressourcen	Sie können jetzt zusätzliche IAM-Ressourcen mit einem Tag-Schlüssel/Wert-Paar kennzeichnen.	11. Februar 2021
StandardPasswortrichtlinie für IAM-Benutzer	Wenn Sie keine benutzerdefinierte Kennwortrichtlinie für Ihre IAM-Benutzerkennwörter festlegen AWS-Konto, müssen sie jetzt der AWS Standard-Passwortrichtlinie entsprechen.	18. November 2020
Die Seiten mit Aktionen, Ressourcen und Bedingungsschlüsseln für AWS Dienste wurden verschoben	Jeder AWS Dienst kann Aktionen, Ressourcen und Bedingungskontextschlüssel für die Verwendung in IAM-Richtlinien definieren. Die Liste der AWS Dienste und ihrer Aktionen, Ressourcen und Bedingungskontextschlüssel finden Sie jetzt in der Service Authorization Reference.	16. November 2020

IAM-Benutzer längere Rollensitzungsdauer	IAM-Benutzer können jetzt eine längere Rollensit zungsdauer haben, wenn sie die Rollen in der wechseln AWS Management Console, wodurch Unterbrechungen aufgrund von Sitzungsa bläufen reduziert werden. Benutzern wird die für die Rolle festgelegte maximale Sitzungsdauer oder die verbleibende Zeit in der Sitzung des IAM-Benutzers gewährt, je nachdem, welcher Wert geringer ist.	24. Juli 2020
Verwenden Sie Service Quotas, um schnelle Erhöhungen für IAM-Entitäten zu beantragen	Über die Konsole Service Quotas können Sie Quotenerh öhungen für anpassbare IAM-Quoten beantragen. Jetzt werden einige Erhöhungen automatisch in der Service Quotas genehmigt und sind innerhalb weniger Minuten in Ihrem Konto verfügbar. Größere Anfragen werden an gesendet. AWS Support	25. Juni 2020

Informationen zum letzten Zugriff in IAM beinhalten jetzt auch Amazon S3-Managementaktionen	Zusätzlich zu den Informationen über den letzten Service-Zugriff können Sie jetzt in der IAM-Konsole Informationen darüber anzeigen, wann ein IAM-Auftraggeber zuletzt eine Amazon S3-Aktion verwendet hat. Sie können auch die AWS API AWS CLI oder verwenden, um den Datenbericht abzurufen. Der Bericht enthält Informationen über die zulässigen Services und Aktionen, auf die Auftraggeber zuletzt zuzugreifen versucht haben, und wann dies geschah. Mit diesen Informationen können Sie unnötige Berechtigungen identifizieren, sodass Sie die IAM-Richtlinien besser an die Regel der geringsten Rechte anpassen können.	3. Juni 2020
Ergänzung zum Sicherheitskapitel	Das Kapitel Sicherheit hilft Ihnen zu verstehen, wie Sie IAM konfigurieren und wie AWS STS Sie Ihre Sicherheits- und Compliance-Ziele erreichen können. Sie erfahren außerdem, wie Sie andere AWS -Services verwenden, um Ihre IAM-Ressourcen zu überwachen und zu schützen.	29. April 2020

sts: Name RoleSession	Sie können jetzt eine Richtlinie schreiben, die Berechtigungen basierend auf dem Sitzungsn amen erteilt, den ein Auftragge ber bei der Übernahme einer Rolle angibt.	21. April 2020
AWS Aktualisierung der Anmeldeseite	Wenn Sie sich auf der AWS Haupt-Anmeldeseite anmelden, können Sie nun wählen, ob Sie sich als IAM-Benutzer Root-Benutzer des AWS-Kontos oder als IAM-Benutzer anmelden möchten. Wenn Sie dies tun, gibt die Bezeichnung auf der Seite an, ob Sie Ihre Root-Benu tzer-E-Mail-Adresse oder Ihre IAM-Benutzerkonto-Informationen angeben sollen. Diese Dokumentation enthält aktualisierte Bildschir maufnahmen, um Ihnen das Verständnis der AWS -Anme ldeseiten zu erleichtern.	4. März 2020

AWS:via AWSService und aws: Bedingungsschlüssel CalledVia	Sie können jetzt eine Richtlinie schreiben, um zu beschränken, ob Services Anforderungen im Auftrag eines IAM-Auftraggebers (Benutzer oder Rolle) ausgeben können. Wenn ein Auftraggeber eine Anforderung an einen AWS -Service ausgibt, kann dieser Service die Anmeldeinformationen des Auftraggebers verwenden, um nachfolgende Anforderungen an andere Services auszugeben. Verwenden Sie den Bedingungsschlüssel `aws:ViaAWSService` , um abzugleichen, ob ein Service eine Anforderung unter Verwendung der Anmeldeinformationen eines Auftraggebers ausgibt. Verwenden Sie die `aws:CalledVia` -Bedingungsschlüssel, um abzugleichen, ob bestimmte Services eine Anforderung mit den Anmeldeinformationen eines Auftraggebers ausgeben.	20. Februar 2020
Der Richtliniensimulator fügt Unterstützung für Berechtigungsgrenzen hinzu	Mit dem IAM-Richtliniensimulator können Sie nun die Auswirkungen von Berechtigungsgrenzen auf IAM-Entitäten testen.	23. Januar 2020

Kontoübergreifende Richtlinienauswertung	Sie können jetzt erfahren, wie Richtlinien für AWS den kontoübergreifenden Zugriff bewertet werden. Dies geschieht, wenn eine Ressource in einem vertrauenswürdigen Konto eine ressourcenbasierte Richtlinie enthält, die einem Auftragge ber in einem anderen Konto den Zugriff auf die Ressource erlaubt. Die Anforderung muss in beiden Konten gewährt werden.	2. Januar 2020
Sitzungs-Tags	Sie können nun Tags einschlie ßen, wenn Sie eine Rolle übernehmen oder einen Benutzer in AWS STS in einen Verbund aufnehmen möchten. Wenn Sie die `GetFedera tionToken` - oder die `AssumeRole` -Operation ausführen, können Sie die Sitzungs-Tags als Attribute übergeben. Wenn Sie die `AssumeRoleWithWebI dentity` Operationen `AssumeRoleWithSAML` oder ausführen, können Sie Attribute aus Ihren Unternehm ensidentitäten an übergeben. AWS	22. November 2019

Steuern Sie den Zugriff für Gruppen von AWS-Konten AWS Organizations	Sie können jetzt AWS Organizations in den IAM-Richtlinien auf Organisationseinheiten (OUs) verweisen. Wenn Sie Organisationen verwenden, um Ihre Konten in Organisationseinheiten zu organisieren, können Sie verlangen, dass Auftragnehmer zu einer bestimmten Organisationseinheit gehören, bevor Sie Zugriff auf Ihre Ressourcen gewähren. Zu den Prinzipalen gehören Root-Benutzer des AWS-Kontos, IAM-Benutzer und IAM-Rollen. Geben Sie dazu den Pfad der Organisationseinheit im Bedingungsschlüssel `aws:PrincipalOrgPaths` in Ihren Richtlinien an.	20. November 2019
Zuletzt verwendete Rolle	Sie können nun das Datum, die Uhrzeit und die Region der letzten Verwendung einer Rolle anzeigen. Diese Informationen helfen Ihnen auch, nicht verwendete Rollen in Ihrem Konto zu identifizieren. Sie können die AWS API AWS CLI und verwenden AWS Management Console, um Informationen darüber einzusehen, wann eine Rolle zuletzt verwendet wurde.	19. November 2019

Aktualisierung der Seite globaler Bedingungskontextschlüssel	Sie können jetzt erfahren, wann die einzelnen globalen Bedingungsschlüssel im Kontext einer Anforderung enthalten sind. Über die Inhaltsverzeichnisseite (TOC) können Sie einfacher zu den einzelnen Schlüsseln navigieren. Die Informationen auf der Seite helfen Ihnen, genauere Richtlinien zu schreiben. Wenn Ihre Mitarbeiter beispielsweise Federation mit IAM-Rollen verwenden, sollten Sie den `aws:userId` -Schlüssel und nicht den Schlüssel `aws:userName` verwenden. Der `aws:userName` -Schlüssel gilt nur für IAM-Benutzer, nicht für Rollen.	6. Oktober 2019

ABAC in AWS	Erfahren Sie, wie die attributebasierte Zugriffsk ontrolle (ABAC) bei der AWS Verwendung von Tags funktioniert und wie sie im Vergleich zum herkömmli chen Autorisierungsmodell abschneidet. AWS Verwenden Sie das ABAC-Tutorial, um zu erfahren, wie Sie eine Richtlini e erstellen und testen, die es IAM-Rollen mit Auftraggeber-Tags erlaubt, auf Ressourcen mit übereinstimmenden Tags zuzugreifen. Diese Strategie ermöglicht es Einzelpersonen, nur die AWS Ressourcen einzusehen oder zu bearbeite n, die sie für ihre Arbeit benötigen.	3. Oktober 2019
AWS STS GetAccessKeyInfo Betrieb	Sie können die AWS Zugriffss chlüssel in Ihrem Code überprüfen, um festzuste llen, ob die Schlüssel von einem Konto stammen, das Sie besitzen. Sie können eine Zugriffsschlüssel-ID mithilfe des `aws sts get-acces s-key-info` AWS CLI Befehls oder der `GetAccess KeyInfo` AWS API-Opera tion übergeben.	24. Juli 2019

Anzeigen von Informationen zum letzten Zugriff auf - Services in IAM	Sie können jetzt im AWS OrganizationsBereich der IAM-Konsole die Informationen zum letzten Zugriff auf einen Dienst für eine AWS Organizations Entität oder Richtlinie anzeigen. Sie können den Datenbericht auch über die AWS API AWS CLI oder abrufen. Diese Daten enthalten Informationen darüber, auf welche Services diese Auftraggeber in einem -Konto zuletzt zuzugreifen versuchten und zu welchem Zeitpunkt dies geschah. Mit diesen Informationen können Sie unnötige Berechtigungen identifizieren, sodass Sie die - oder Organizations-Richtlinien besser an die Regel der geringsten Rechte anpassen können.	20. Juni 2019
Verwendung einer verwalteten Richtlinie als Sitzungsrichtlinie	Sie können beim Übernehmen einer Rolle jetzt bis zu 10 verwaltete Richtlinien-ARNs übergeben. Auf diese Weise können Sie die Berechtigungen der temporären Anmeldedaten der Rolle beschränken.	7. Mai 2019

AWS STS Regionskompatibilität der Sitzungstoken für den globalen Endpunkt	Sie können jetzt auswählen, ob globale Endpunkt-Token der Version 1 oder Version 2 verwendet werden sollen. Tokens der Version 1 sind nur in AWS Regionen gültig, die standardmäßig verfügbar sind. Diese Token funktionieren icht in manuell aktivierten Regionen, wie z. B. Asien-Pazifik (Hongkong), nicht. Token der Version 2 sind in allen Regionen gültig. Token der Version 2 sind jedoch länger und können sich auf Systeme auswirken, in denen Sie Token vorübergehend speichern.	26. April 2019
Erlaube das Aktivieren und Deaktivieren von Regionen AWS	Sie können jetzt eine Richtlinie erstellen, mit der ein Administrator die Region Asien-Pazifik (Hongkong) (ap-east-1) aktivieren und deaktivieren kann.	24. April 2019

IAM-Benutzer, Seite Meine Sicherheitsanmeldeinformationen	IAM-Benutzer können jetzt alle ihre eigenen Anmeldeinformationen auf der Seite My Security Credentials (Meine Sicherheitsinformationen) verwalten. AWS Management Console Auf dieser Seite werden Kontoinformationen wie die Konto-ID und die kanonische Benutzer-ID angezeigt. Benutzer können auch ihre eigenen Passwörter, Zugriffsschlüssel, X.509-Zertifikate, SSH-Schlüssel und Git-Anmeldeinformationen anzeigen und bearbeiten.	24. Januar 2019
Zugriff auf Advisor-API	Sie können jetzt die AWS API AWS CLI und verwenden, um Informationen zum zuletzt aufgerufenen Dienst anzuzeigen.	7. Dezember 2018
Taggen von IAM-Benutzern und Rollen	Sie können nun IAM-Tags verwenden, um benutzerdefinierte Attribute zu einer Identität (IAM-Benutzer oder IAM-Rolle) mithilfe eines Tag-Schlüssel/Wert-Paars hinzuzufügen. Sie können auch Tags verwenden, um den Zugriff einer Identität auf Ressourcen zu steuern oder zu kontrollieren, welche Tags mit einer Identität verknüpft werden können.	14. November 2018

U2F-Sicherheitsschlüssel	Sie können nun U2F-Sicherheitsschlüssel als MFA-Option (Multi-Faktor-Authentifizierung) verwenden, wenn Sie sich bei der AWS Management Console anmelden.	25. September 2018
Unterstützung für Amazon VPC-Endpunkte	Sie können jetzt eine private Verbindung zwischen Ihrer VPC und der AWS STS Region USA West (Oregon) herstellen.	31. Juli 2018
Berechtigungsgrenzen	Neue Funktion, mit der vertrauenswürdigen Mitarbeitern die Fähigkeit, IAM-Berechtigungen zu verwalten, leichter gewährt werden kann, ohne gleichzeitig uneingeschränkten administrativen IAM-Zugriff zu gewähren.	12. Juli 2018
aws: ID PrincipalOrg	Der neue Bedingungsschlüssel bietet eine einfachere Möglichkeit, den Zugriff auf AWS Ressourcen zu kontrollieren, indem die AWS Organisation der IAM-Prinzipale angegeben wird.	17. Mai 2018
als: RequestedRegion	Der neue Bedingungsschlüssel bietet eine einfachere Möglichkeit, IAM-Richtlinien zur Steuerung des Zugriffs auf AWS Regionen zu verwenden.	25. April 2018

Erhöhte Sitzungsdauer für IAM-Rollen	Eine IAM-Rolle kann jetzt eine Sitzungsdauer von 12 Stunden haben.	28. März 2018
Aktualisierter Workflow beim Erstellen einer Rolle	Der neue Workflow verbesser t die Erstellung von Vertrauen sbeziehungen und das Hinzufügen von Berechtig ungen zu Rollen.	8. September 2017
AWS-Konto Anmeldevorgang	Dank des aktualisierten AWS Anmeldevorgangs können sowohl Root-Benutzer als auch IAM-Benutzer den Link „Bei der Konsole anmelden“ auf der Startseite AWS Management Console von verwenden.	25. August 2017
IAM-Beispielrichtlinien	Das Dokumentationsupdate enthält mehr als 30 Beispielr ichtlinien.	2. August 2017
IAM Best Practices	Informationen, die dem Abschnitt Users (Benutzer) der IAM Konsole hinzugefügt wurden, machen es einfacher , bewährte Methoden für IAM einzuhalten.	5. Juli 2017
Auto-Scaling-Ressourcen	Berechtigungen auf Ressource nebene können den Zugriff auf und die Berechtigungen für Ressourcen mit Auto Scaling Skalierung steuern.	16. Mai 2017

Amazon RDS for MySQL und Amazon Aurora Datenbanken	Datenbankadministratoren können Datenbankbenutzer IAM-Benutzern und -Rollen zuordnen und so den Benutzerzugriff auf alle AWS Ressourcen von einem zentralen Ort aus verwalten.	24. April 2017
Serviceverknüpfte Rollen	Mit Diensten verknüpfte Rollen bieten eine einfacher e und sicherere Möglichkeit, Berechtigungen an Dienste zu delegieren. AWS	19. April 2017
Richtlinienübersichten	Neue Richtlinienübersichten erleichtern das Verständnis von Berechtigungen in IAM-Richtlinien.	23. März 2017

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.