aws

Benutzerhandbuch

Amazon Verified Permissions

Amazon Verified Permissions: Benutzerhandbuch

Table of Contents

Was ist Amazon Verified Permissions? 1
Autorisierung im Rahmen verifizierter Berechtigungen 1
Sprache der Cedar-Richtlinie 2
Vorteile verifizierter Berechtigungen 2
Beschleunigen Sie die Anwendungsentwicklung 2
Sicherere Anwendungen 3
Funktionen für Endbenutzer 3
Zugehörige Services 3
Zugriff auf verifizierte Berechtigungen 3
Preise für verifizierte Berechtigungen 5
Erste Schritte mit Policy Stores 7
Voraussetzungen 8
Schritt 1: Erstellen Sie einen Richtlinienspeicher PhotoFlash 10
Schritt 2: Erstellen Sie eine Richtlinie 11
Schritt 3: Testen eines Richtlinienspeichers 12
Schritt 4: Bereinigen von Ressourcen 13
Entwerfen eines Autorisierungsmodells 15
Kein einziges richtiges Modell 16
Fehler zurücksenden 17
Konzentrieren Sie sich auf Ressourcen 17
Ziehen Sie Mehrmandantenverträge in Betracht 19
Vergleich von gemeinsam genutzten Richtlinienspeichern und Richtlinienspeichern pro Mandant 20
Wie soll man wählen 22
Richtlinien werden gespeichert 23
Richtlinienspeicher erstellen 24
Einen Policy Store mit Rust erstellen 33
API-verknüpfte Richtlinienspeicher 38
Funktionsweise 40
Überlegungen 42
ABAC hinzufügen 43
Übergang zur Produktion 44
Fehlerbehebung 47
Löschen von Richtlinienspeichern 50

Schema des Richtlinienspeichers 52
Schema bearbeiten 54
Modus zur Überprüfung der Richtlinien 58
Richtlinien 60
Statische Richtlinien erstellen 61
Statische Richtlinien bearbeiten 63
..... 65
Evaluieren Sie den Beispielkontext 68
Testen von -Richtlinien 73
Beispielrichtlinien 76
Verwendet die Klammernotation, um auf Token-Attribute zu verweisen 77
Verwendet Punktnotation, um auf Attribute zu verweisen 77
Spiegelt Amazon Cognito Cognito-ID-Token-Attribute wider 77
Spiegelt die OIDC-ID-Token-Attribute wider 78
Spiegelt die Attribute des Amazon Cognito Cognito-Zugriffstokens wider 78
Spiegelt die Attribute von OIDC-Zugriffstoken wider 79
Richtlinienvorlagen und mit Vorlagen verknüpfte Richtlinien 80
Richtlinienvorlagen erstellen 81
Richtlinien erstellen, die mit Vorlagen verknüpft sind 82
Richtlinienvorlagen bearbeiten 84
Beispiel für Richtlinien, die mit Vorlagen verknüpft sind 86
Beispiele für PhotoFlash 86
DigitalPetStore Beispiele 88
Beispiele für TinyToDo 88
Identitätsquellen 90
Arbeiten mit Amazon Cognito Cognito-Identitätsquellen 91
Arbeiten mit OIDC-Identitätsquellen 93
Validierung von Kunden und Zielgruppen 95
Kundenseitige Autorisierung für JWTs 96
Identitätsquellen erstellen 98
Amazon Cognito Cognito-Identitätsquelle 99
OIDC-Identitätsquelle 102
Identitätsquellen bearbeiten 105
Identitätsquelle für Amazon Cognito Cognito-Benutzerpools 105
OpenID Connect (OIDC) -Identitätsquelle 107
Zuordnen von Tokens zum Schema 109

Zuordnen von ID-Token ... 110
Zugriffstoken zuordnen ... 114
Alternative Schreibweise für durch Doppelpunkte getrennte Amazon Cognito-Ansprüche ... 119
Wissenswertes über Schema-Mapping ... 120
Integrationen ... 125
Express verwenden ... 125
Voraussetzungen ... 126
Einrichtung der Integration ... 126
Autorisierung konfigurieren ... 127
Implementierung der Autorisierungs-Middleware ... 130
Die Integration testen ... 131
Fehlerbehebung ... 131
Nächste Schritte ... 131
Anfragen autorisieren ... 132
API-Operationen ... 133
Modell testen ... 134
Integration mit Anwendungen ... 136
Sicherheit ... 139
Datenschutz ... 139
Datenverschlüsselung ... 141
Identity and Access Management ... 141
Zielgruppe ... 142
Authentifizierung mit Identitäten ... 143
Verwalten des Zugriffs mit Richtlinien ... 146
So funktioniert Amazon Verified Permissions mit IAM ... 149
IAM Richtlinien für verifizierte Berechtigungen ... 156
Beispiele für identitätsbasierte Richtlinien ... 159
AWS verwaltete Richtlinien ... 163
Fehlerbehebung ... 166
Compliance-Validierung ... 168
Ausfallsicherheit ... 170
Überwachen ... 171
CloudTrail protokolliert ... 171
Informationen zu verifizierten Berechtigungen in CloudTrail ... 172
Grundlegendes zu Einträgen in der Protokolldatei „Verifizierte ... 173
Arbeiten mit AWS CloudFormation ... 191

Verifizierte Berechtigungen und AWS CloudFormation Vorlagen ... 191
AWS CDK-Konstrukte ... 192
Erfahren Sie mehr über AWS CloudFormation ... 192
Verwenden AWS PrivateLink ... 193
Überlegungen ... 193
Erstellen eines Schnittstellenendpunkts ... 193
Erstellen einer Endpunktrichtlinie ... 194
Kontingente ... 196
Kontingente für Ressourcen ... 196
Beispiel für die Größe einer Richtlinie, die mit einer Vorlage verknüpft ist ... 197
Kontingente für Hierarchien ... 199
Kontingente für Operationen pro Sekunde ... 200
Begriffe und Konzepte ... 205
Autorisierungsmodell ... 206
Autorisierungsanfrage ... 206
Antwort auf die Autorisierung ... 206
Überlegte Richtlinien ... 207
Kontextdaten ... 207
Festlegung von Richtlinien ... 207
Daten der Entität ... 207
Berechtigungen, Autorisierung und Prinzipale ... 207
Durchsetzung von Richtlinien ... 208
Richtlinienspeicher ... 208
Zufriedene Richtlinien ... 208
Unterschiede zu Cedar ... 208
Namespace-Definition ... 209
Unterstützung von Richtlinienvorlagen ... 209
Schema-Unterstützung ... 209
Definition von Aktionsgruppen ... 209
Formatierung von Entitäten ... 210
Längen- und Größenbeschränkungen ... 215
Häufig gestellte Fragen ... 217
Wie ist der aktuelle Stand des Upgrades? ... 217
Muss ich jetzt etwas tun? ... 217
Wirkt sich das Upgrade der Konsole auf den Autorisierungsdienst aus? ... 217
Was sind die wichtigsten Änderungen in Cedar v3 und Cedar v4? ... 218

Wann wird das Upgrade auf Cedar v4 abgeschlossen sein? 218
Dokumentverlauf 219
..... ccxxi

Was ist Amazon Verified Permissions?

Amazon Verified Permissions ist ein skalierbarer, detaillierter Berechtigungsverwaltungs- und Autorisierungsservice für benutzerdefinierte Anwendungen, die von Ihnen erstellt wurden. Verified Permissions ermöglicht es Ihren Entwicklern, sichere Anwendungen schneller zu erstellen, indem die Autorisierung externalisiert und die Richtlinienverwaltung und -verwaltung zentralisiert wird. Verified Permissions verwendet die Cedar-Richtliniensprache, um detaillierte Berechtigungen zum Schutz der Ressourcen Ihrer Anwendung zu definieren.

Anleitungen und Beispiele für die Einrichtung eines Policy Decision Point (PDP) mithilfe verifizierter Berechtigungen finden Sie unter Implementieren eines PDP mithilfe von Amazon Verified Permissions in AWS Prescriptive Guidance.

Themen

- Autorisierung im Rahmen verifizierter Berechtigungen
- Sprache der Cedar-Richtlinie
- Vorteile verifizierter Berechtigungen
- Zugehörige Services
- Zugriff auf verifizierte Berechtigungen
- Preise für verifizierte Berechtigungen

Autorisierung im Rahmen verifizierter Berechtigungen

Verified Permissions ermöglicht die Autorisierung, indem überprüft wird, ob ein Principal in einem bestimmten Kontext in Ihrer Anwendung eine Aktion an einer Ressource ausführen darf. Verified Permissions geht davon aus, dass der Principal zuvor auf andere Weise identifiziert und authentifiziert wurde, z. B. mithilfe von Protokollen wie OpenID Connect, einem gehosteten Anbieter wie Amazon Cognito oder einer anderen Authentifizierungslösung. Verified Permissions ist unabhängig davon, wo der Principal verwaltet wird und wie er authentifiziert wurde.

Verified Permissions ist ein Service, mit dem Kunden Richtlinien in Form von Verified Permissions programmgesteuert oder mithilfe von AWS Management Console Infrastructure-as-Code-Lösungen wie Infrastructure as Code erstellen APIs, verwalten und testen können. AWS CloudFormation Berechtigungen werden in der Cedar-Richtliniensprache ausgedrückt. Die Client-Anwendung ruft die

Autorisierung APIs auf, um die im Service gespeicherten Cedar-Richtlinien auszuwerten und eine Zugriffsentscheidung darüber zu treffen, ob eine Aktion zulässig ist.

Sprache der Cedar-Richtlinie

Die Autorisierungsrichtlinien in Verified Permissions werden in der Cedar-Richtliniensprache geschrieben. Cedar ist eine Open-Source-Sprache zum Schreiben von Autorisierungsrichtlinien und zum Treffen von Autorisierungsentscheidungen auf der Grundlage dieser Richtlinien. Wenn Sie eine Anwendung erstellen, müssen Sie sicherstellen, dass nur autorisierte Benutzer oder Maschinen auf die Anwendung zugreifen können und nur das tun können, wozu sie autorisiert sind. Mit Cedar können Sie Ihre Geschäftslogik von der Autorisierungslogik entkoppeln. Im Code Ihrer Anwendung stellen Sie Anfragen, die an Ihre Operationen gestellt werden, einen Aufruf an die Cedar-Autorisierungs-Engine mit der Frage „Ist diese Anfrage autorisiert?“ voran. Anschließend kann die Anwendung entweder den angeforderten Vorgang ausführen, wenn die Entscheidung „Zulassen“ lautet, oder eine Fehlermeldung zurückgeben, wenn die Entscheidung „Verweigern“ lautet.

Verified Permissions verwendet derzeit Cedar Version 2.4.

Weitere Informationen zu Cedar finden Sie im Folgenden:

- Referenzhandbuch zur Sprachenpolitik von Cedar
- Cedar GitHub Repository

Vorteile verifizierter Berechtigungen

Beschleunigen Sie die Anwendungsentwicklung

Beschleunigen Sie die Anwendungsentwicklung, indem Sie die Autorisierung von der Geschäftslogik entkoppeln.

Verified Permissions bietet Integrationen mit gängigen Entwicklungs-Frameworks und erleichtert so die Implementierung der Autorisierung in Ihren Anwendungen mit minimalen Codeänderungen. Diese Integrationen ermöglichen es Ihnen, sich auf Ihre Kerngeschäftslogik zu konzentrieren, während Verified Permissions die Autorisierungsentscheidungen übernimmt.

- Express.js — Eine Middleware-basierte Integration, mit der Sie API-Endpunkte in Ihren Express-Anwendungen schützen können, ohne bestehende Route-Handler ändern zu müssen. Weitere Informationen finden Sie unter the section called “Express verwenden”.

Sicherere Anwendungen

Verified Permissions ermöglicht es Entwicklern, sicherere Anwendungen zu erstellen.

Funktionen für Endbenutzer

Mit Verified Permissions können Sie umfassendere Funktionen für die Verwaltung von Berechtigungen für Endbenutzer bereitstellen.

Zugehörige Services

- Amazon Cognito — Amazon Cognito ist eine Identitätsplattform für Web- und mobile Apps. Es ist ein Benutzerverzeichnis, ein Authentifizierungsserver und ein Autorisierungsdienst für OAuth 2.0-Zugriffstoken und AWS Anmeldeinformationen. Wenn Sie einen Richtlinienspeicher erstellen, haben Sie die Möglichkeit, Ihre Prinzipale und Gruppen aus einem Amazon Cognito Cognito-Benutzerpool zu erstellen. Weitere Informationen finden Sie im Amazon Cognito Entwicklerhandbuch.
- Amazon API Gateway — Amazon API Gateway ist ein AWS Service für die Erstellung, Veröffentlichung, Wartung, Überwachung und Sicherung von REST, HTTP und WebSocket APIs in jeder Größenordnung. Wenn Sie einen Richtlinienspeicher erstellen, haben Sie die Möglichkeit, Ihre Aktionen und Ressourcen aus einer API in API Gateway zu erstellen. Weitere Informationen zu API Gateway finden Sie im API Gateway Developer Guide.
- AWS IAM Identity Center— Mit IAM Identity Center können Sie die Anmeldesicherheit für Ihre Mitarbeiteridentitäten, auch Workforce-Benutzer genannt, verwalten. IAM Identity Center bietet einen zentralen Ort, an dem Sie Workforce-Benutzer erstellen oder verbinden und ihren Zugriff auf all ihre Anwendungen zentral verwalten können. AWS-Konten Weitere Informationen finden Sie im AWS IAM Identity Center -Benutzerhandbuch.

Zugriff auf verifizierte Berechtigungen

Sie können mit Amazon Verified Permissions auf eine der folgenden Arten arbeiten.

AWS Management Console

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

- Konsole „Amazon Verified Permissions“

AWS Tools für die Befehlszeile

Sie können die AWS Befehlszeilentools verwenden, um Befehle an der Befehlszeile Ihres Systems auszugeben, um verifizierte Berechtigungen und AWS Aufgaben auszuführen. Die Verwendung der Kommandozeile kann schneller und bequemer sein als die Konsole. Die Befehlszeilen-Tools können auch beim Erstellen von Skripts für AWS -Aufgaben hilfreich sein.

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.

- verifiedpermissions in der Befehlsreferenz AWS CLI
- Von Amazon verifizierte Berechtigungen in AWS Tools for Windows PowerShell

AWS SDKs

AWS stellt SDKs (Software Development Kits) bereit, die aus Bibliotheken und Beispielcode für verschiedene Programmiersprachen und Plattformen (Java, Python, Ruby, .NET, iOS, Android usw.) bestehen. SDKs Sie bieten eine bequeme Möglichkeit, programmatischen Zugriff auf verifizierte Berechtigungen und AWS zu erstellen. SDKs Sie kümmern sich beispielsweise um Aufgaben wie das kryptografische Signieren von Anfragen, das Verwalten von Fehlern und das automatische Wiederholen von Anfragen.

Weitere Informationen und Downloads finden Sie unter AWS SDKs Tools für. Amazon Web Services

Im Folgenden finden Sie Links zur Dokumentation für Ressourcen zu verifizierten Berechtigungen in verschiedenen Bereichen AWS SDKs.

- AWS SDK für .NET
- AWS SDK für C++
- AWS SDK für Go
- AWS SDK für Java
- AWS SDK für JavaScript
- AWS SDK für PHP
- AWS SDK for Python (Boto)

- AWS SDK für Ruby
- AWS SDK for Rust

AWS CDK-Konstrukte

Das AWS Cloud Development Kit (AWS CDK) ist ein Open-Source-Framework für die Softwareentwicklung, mit dem Cloud-Infrastruktur im Code definiert und bereitgestellt werden kann. AWS CloudFormation Konstrukte oder wiederverwendbare Cloud-Komponenten können zum Erstellen von Vorlagen verwendet werden. AWS CloudFormation Diese Vorlagen können dann zur Bereitstellung Ihrer Cloud-Infrastruktur verwendet werden.

Weitere Informationen und das Herunterladen von AWS CDK finden Sie unter AWS Cloud Development Kit.

Im Folgenden finden Sie Links zur Dokumentation für AWS CDK Ressourcen mit verifizierten Berechtigungen, z. B. Konstrukte.

- Von Amazon verifizierte Berechtigungen L2 CDK Construct

API für verifizierte Berechtigungen

Sie können auf verifizierte Berechtigungen und AWS programmgesteuert zugreifen, indem Sie die Verified Permissions API verwenden, mit der Sie HTTPS-Anfragen direkt an den Dienst senden können. Wenn Sie die -API nutzen, müssen Sie Code zur digitalen Signierung von Anfragen mittels Ihrer Anmeldeinformationen einschließen.

- Referenzhandbuch zur Amazon Verified Permissions API

Preise für verifizierte Berechtigungen

Verified Permissions bietet gestaffelte Preise, die auf der Anzahl der Autorisierungsanfragen pro Monat basieren, die Ihre Anträge bei Verified Permissions stellen. Es gibt auch Preise für Richtlinienverwaltungsaktionen, die auf der Anzahl der cURL-Policy-API-Anfragen (Client-URL) pro Monat basieren, die von Ihren Anwendungen an Verified Permissions gestellt werden.

Eine vollständige Liste der Gebühren und Preise für verifizierte Berechtigungen finden Sie unter Amazon Verified Permissions — Preise.

Um Ihre Rechnung anzuzeigen, navigieren Sie zu Fakturierungs- und Kostenverwaltungs-Dashboard in der AWS Fakturierung und Kostenmanagement -Konsole. Ihre Abrechnung enthält Links zu Nutzungsberichten mit Details zu Ihrer Abrechnung. Weitere Informationen zur AWS-Konto Abrechnung finden Sie im AWS Billing Benutzerhandbuch.

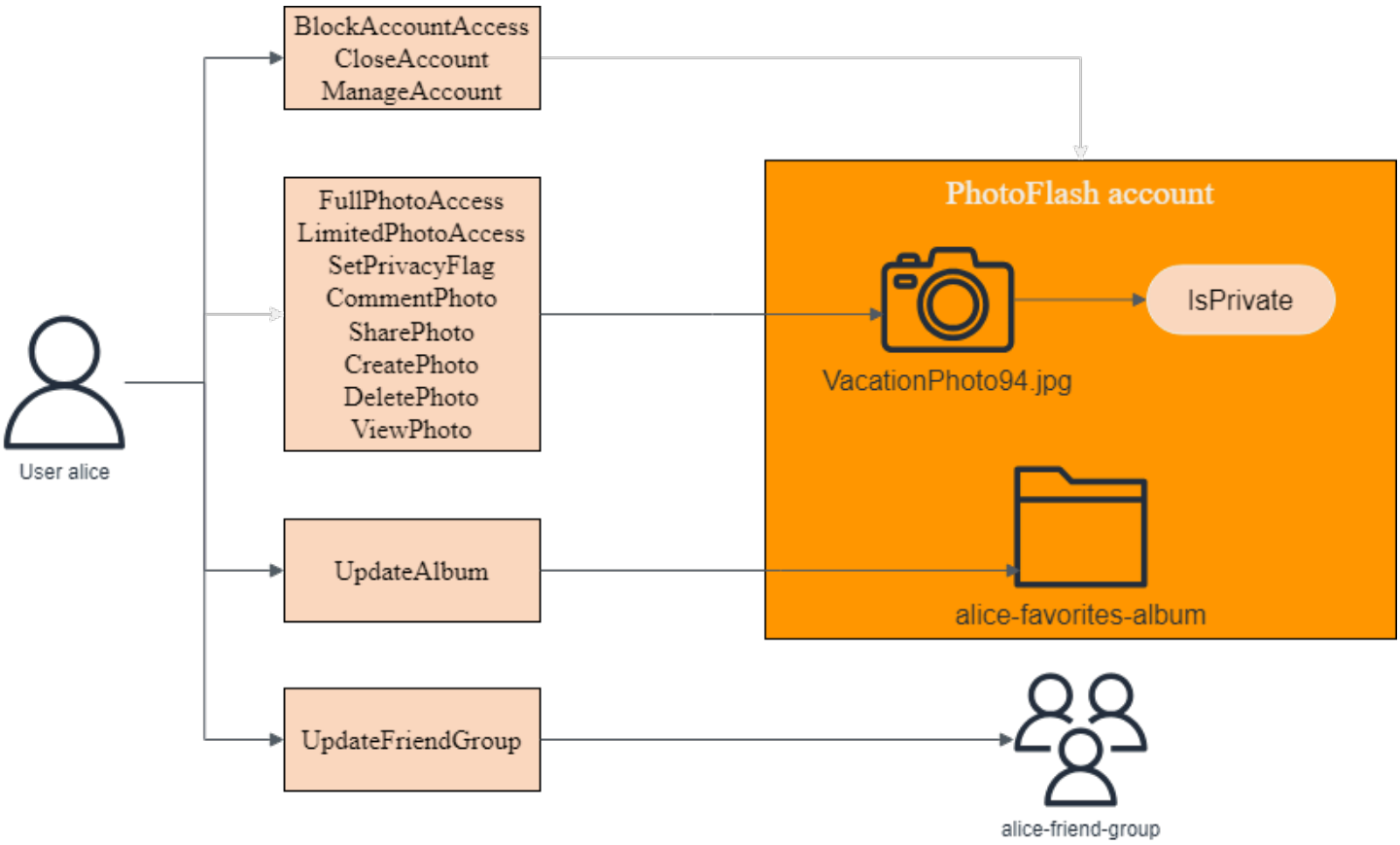
Wenn Sie Fragen zu AWS Abrechnung, Konten und Veranstaltungen haben, wenden Sie sich an Support.

Erstellen Sie Ihren ersten Amazon Verified Permissions Policy Store

Gehen wir für dieses Tutorial davon aus, dass Sie der Entwickler einer Anwendung zum Teilen von Fotos sind und nach einer Möglichkeit suchen, zu kontrollieren, welche Aktionen die Benutzer der Anwendung ausführen können. Sie möchten steuern, wer Fotos und Fotoalben hinzufügen, löschen oder ansehen kann. Sie möchten auch kontrollieren, welche Aktionen ein Benutzer auf seinem Konto ausführen kann. Können sie ihr Konto verwalten, wie wäre es mit dem Konto eines Freundes? Um diese Aktionen zu kontrollieren, müssten Sie Richtlinien erstellen, die diese Aktionen auf der Grundlage der Identität des Benutzers zulassen oder verbieten. Verified Permissions bietet Richtlinienspeicher oder Container, in denen diese Richtlinien gespeichert werden.

In diesem Tutorial erfahren Sie, wie Sie mithilfe der Amazon Verified Permissions-Konsole einen Muster-Policy-Shop erstellen. Die Konsole bietet einige Beispieloptionen für den Policy Store, und wir werden einen PhotoFlashPolicy Store erstellen. Dieser Richtlinienspeicher ermöglicht es Prinzipalen, z. B. Benutzern, Aktionen wie das Teilen von Ressourcen wie Fotos oder Alben durchzuführen.

Das folgende Diagramm veranschaulicht die Beziehungen zwischen einer Principal und die AktionenUser::alice, die sie für verschiedene Ressourcen ausführen kann, nämlich für ihr PhotoFlash Konto, die VactionPhoto94.jpg Datei, das Fotoalbum alice-favorites-album und die Benutzergruppealice-friend-group.



Nachdem Sie sich mit dem PhotoFlashRichtlinienspeicher vertraut gemacht haben, wollen wir den Richtlinienspeicher erstellen und ihn näher untersuchen.

Voraussetzungen

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/die Anmeldung.
2. Folgen Sie den Online-Anweisungen.

   Ein Teil des Anmeldevorgangs umfasst den Empfang eines Telefonanrufs oder einer Textnachricht und die Eingabe eines Bestätigungscodes auf der Telefontastatur.

   Wenn Sie sich für eine anmelden AWS-Konto, wird eine Root-Benutzer des AWS-Kontoserstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Als bewährte

Sicherheitsmethode weisen Sie einem Administratorbenutzer Administratorzugriff zu und verwenden Sie nur den Root-Benutzer, um Aufgaben auszuführen, die Root-Benutzerzugriff erfordern.

AWS sendet Ihnen nach Abschluss des Anmeldevorgangs eine Bestätigungs-E-Mail. Sie können Ihre aktuellen Kontoaktivitäten jederzeit einsehen und Ihr Konto verwalten, indem Sie zu https://aws.amazon.com/gehen und Mein Konto auswählen.

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

Erstellen eines Benutzers 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 Administratorbenutzer im IAM Identity Center Benutzerzugriff.

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

Anmelden als Administratorbenutzer

- 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.

Weiteren Benutzern Zugriff zuweisen

1. Erstellen Sie im IAM-Identity-Center einen Berechtigungssatz, der den bewährten Vorgehensweisen für die Anwendung von geringsten Berechtigungen folgt.

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

   Eine genaue Anleitung finden Sie unter Gruppen hinzufügen im AWS IAM Identity Center Benutzerhandbuch.

Schritt 1: Erstellen Sie einen Richtlinienspeicher PhotoFlash

Im folgenden Verfahren erstellen Sie mithilfe der AWS Konsole einen PhotoFlashRichtlinienspeicher.

Um einen PhotoFlash Richtlinienspeicher zu erstellen

1. Wählen Sie in der Konsole „Verifizierte Berechtigungen“ die Option Neuen Richtlinienspeicher erstellen aus.
2. Wählen Sie für Startoptionen die Option Aus einem Beispielrichtlinienspeicher starten aus.
3. Wählen Sie für Beispielprojekt die Option PhotoFlash.
4. Wählen Sie „Richtlinienspeicher erstellen“ aus.

Sobald Sie die Meldung „Richtlinienspeicher erstellt und konfiguriert“ sehen, wählen Sie Gehe zur Übersicht, um Ihren Richtlinienspeicher zu erkunden.

Schritt 2: Erstellen Sie eine Richtlinie

Bei der Erstellung des Richtlinienspeichers wurde eine Standardrichtlinie erstellt, die es Benutzern ermöglicht, die volle Kontrolle über ihre eigenen Konten zu haben. Dies ist eine nützliche Richtlinie, aber für unsere Zwecke sollten wir eine restriktivere Richtlinie erstellen, um die Nuancen verifizierter Berechtigungen zu untersuchen. Wenn Sie sich an das Diagramm erinnern, das wir uns zu Beginn des Tutorials angesehen haben, hatten wir einen Principal`User::alice`, der eine Aktion,`UpdateAlbum`, an einer Ressource, ausführen konnte`alice-favorites-album`. Fügen wir die Richtlinie hinzu, die es Alice und nur Alice erlaubt, dieses Album zu verwalten.

Um eine Richtlinie zu erstellen

1. Wählen Sie in der Konsole „Verifizierte Berechtigungen“ den Richtlinienspeicher aus, den Sie in Schritt 1 erstellt haben.
2. Wählen Sie in der Navigation die Option Richtlinien aus.
3. Wählen Sie Richtlinie erstellen und dann Statische Richtlinie erstellen aus.
4. Wählen Sie für Richtlinieneffekt die Option Zulassen aus.
5. Wählen Sie für Principals scope die Option Specific Principal aus, wählen Sie dann für Entitätstyp angeben die Option PhotoFlash: :User aus und geben Sie für Entitäts-ID angeben den Wert ein. **`alice`**
6. Wählen Sie für Ressourcenbereich die Option Spezifische Ressource aus, wählen Sie dann für Entitätstyp angeben die Option PhotoFlash: :Album und geben Sie für Entitäts-ID angeben den Wert ein. **`alice-favorites-album`**
7. Wählen Sie für Aktionsbereich die Option Spezifische Gruppe von Aktionen und dann für Aktion (en), für die diese Richtlinie gelten soll, die Option aus UpdateAlbum.
8. Wählen Sie Weiter aus.
9. Geben Sie unter Details für Richtlinienbeschreibung — optional Folgendes ein**`Policy allowing alice to update alice-favorites-album.`**.
10. Wählen Sie Richtlinie erstellen aus

Nachdem Sie eine Richtlinie erstellt haben, können Sie sie in der Konsole „Verifizierte Berechtigungen“ testen.

Schritt 3: Testen eines Richtlinienspeichers

Nachdem Sie Ihren Richtlinienspeicher und Ihre Richtlinie erstellt haben, können Sie sie testen, indem Sie eine simulierte Autorisierungsanfrage mit dem Prüfstand für verifizierte Berechtigungen ausführen.

Um Richtlinien des Richtlinienspeichers zu testen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite die Option Testbench aus.
3. Wählen Sie den Visuellen Modus.
4. Gehen Sie für Principal wie folgt vor:
    a. Wählen Sie für Principal, der eine Aktion PhotoFlash durchführt: :User und für Entitätsbezeichner angeben den **`alice`** Wert ein.
    b. Stellen Sie unter Attribute für Account: Entität sicher, dass die Entität PhotoFlash: :Account ausgewählt ist, und geben Sie für Entitäts-ID angeben den Wert ein. **`alice-account`**
5. Wählen Sie unter Ressource für Ressource, auf die der Prinzipal reagiert, den Ressourcentyp PhotoFlash: :Album und geben Sie für Entitäts-ID angeben den Wert ein. **`alice-favorites-album`**
6. Wählen Sie für Aktion PhotoFlash: :Action::“ UpdateAlbum "aus der Liste der gültigen Aktionen aus.
7. Wählen Sie oben auf der Seite die Option Autorisierungsanfrage ausführen aus, um die Autorisierungsanfrage für die Cedar-Richtlinien im Beispielrichtlinienspeicher zu simulieren. Auf dem Prüfstand sollte Entscheidung: Zulassen angezeigt werden, was bedeutet, dass unsere Richtlinie erwartungsgemäß funktioniert.

Die folgende Tabelle enthält zusätzliche Werte für den Prinzipal, die Ressource und die Aktion, die Sie mit dem Prüfstand für verifizierte Berechtigungen testen können. Die Tabelle enthält die Entscheidung über die Autorisierungsanfrage, die auf den statischen Richtlinien basiert, die im PhotoFlash Beispielrichtlinienspeicher enthalten sind, und auf der Richtlinie, die Sie in Schritt 2 erstellt haben.

Hauptwert	Hauptkonto: Unternehmenswert	Wert der Ressource	Wert der übergeordneten Ressource	Action (Aktion)	Entscheidung über die Autorisierung
PhotoFlash: :Benutzer | bob	PhotoFlash: :Konto | Alice-Konto	PhotoFlash:: Album | alice-favorites-album	N/A	PhotoFlash: :Aktion:: “ "UpdateAlbum	Deny
PhotoFlash: :Benutzer | alice	PhotoFlash: :Konto | Alice-Konto	PhotoFlash: :Foto | photo.jpeg	PhotoFlash: :Konto | Bob-Konto	PhotoFlash: :Aktion::“ "ViewPhoto	Deny
PhotoFlash: :Benutzer | alice	PhotoFlash: :Konto | Alice-Konto	PhotoFlash: :Foto | photo.jpeg	PhotoFlash: :Konto | Alice-Konto	PhotoFlash: :Aktion::“ "ViewPhoto	Sobald Sie die Details auf dieser Seite überprüft haben, klicken Sie auf
PhotoFlash: :Benutzer | alice	PhotoFlash: :Konto | Alice-Konto	PhotoFlash: :Foto | bob-photo.jpeg	PhotoFlash:: Album | Bob-Vacation-Album	PhotoFlash: :Aktion::“ "DeletePhoto	Deny

Schritt 4: Bereinigen von Ressourcen

Nachdem Sie Ihren Richtlinienspeicher durchsucht haben, löschen Sie ihn.

Um einen Richtlinienspeicher zu löschen

1. Wählen Sie in der Konsole „Verifizierte Berechtigungen“ den Richtlinienspeicher aus, den Sie in Schritt 1 erstellt haben.
2. Wählen Sie in der Navigation Einstellungen aus.

3. Wählen Sie unter Richtlinienspeicher löschen die Option Diesen Richtlinienspeicher löschen aus.
4. Im Bereich Diesen Richtlinienspeicher löschen? Geben Sie im Dialogfeld Löschen ein, und wählen Sie dann Löschen aus.

Bewährte Methoden für die Gestaltung eines Autorisierungsmodells

Wenn Sie sich darauf vorbereiten, den Service Amazon Verified Permissions in einer Softwareanwendung zu nutzen, kann es schwierig sein, als ersten Schritt sofort mit dem Verfassen von Richtlinienerklärungen zu beginnen. Dies wäre vergleichbar mit dem Beginn der Entwicklung anderer Teile einer Anwendung, indem Sie SQL-Anweisungen oder API-Spezifikationen schreiben, bevor Sie vollständig entscheiden, was die Anwendung tun soll. Stattdessen sollten Sie mit einer Benutzererfahrung beginnen. Arbeiten Sie dann von dieser Erfahrung aus rückwärts, um zu einem Implementierungsansatz zu gelangen.

Während Sie diese Arbeit erledigen, werden Sie feststellen, dass Sie sich Fragen stellen wie:

- Was sind meine Ressourcen? Wie sind sie organisiert? Befinden sich Dateien beispielsweise in einem Ordner?
- Spielt die Organisation der Ressourcen eine Rolle im Berechtigungsmodell?
- Welche Aktionen können Principals für jede Ressource ausführen?
- Wie erwerben Principals diese Berechtigungen?
- Möchten Sie, dass Ihre Endbenutzer aus vordefinierten Berechtigungen wie „Admin“, „Operator“ oder „“ wählen können, oder sollten sie Ad-hoc-Richtlinienerklärungen erstellen? ReadOnly Oder beides?
- Handelt es sich um globale oder bereichsspezifische Rollen? Ist ein „Operator“ beispielsweise auf einen einzelnen Mandanten beschränkt, oder bedeutet „Operator“ Operator für die gesamte Anwendung?
- Welche Arten von Abfragen sind erforderlich, um die Benutzererfahrung zu verbessern? Müssen Sie beispielsweise alle Ressourcen auflisten, auf die ein Principal zugreifen kann, um die Startseite dieses Benutzers zu rendern?
- Können sich Benutzer versehentlich selbst von ihren eigenen Ressourcen ausschließen? Muss das vermieden werden?

Das Endergebnis dieser Übung wird als Autorisierungsmodell bezeichnet. Es definiert die Prinzipien, Ressourcen und Aktionen und wie sie zueinander in Beziehung stehen. Für die Erstellung dieses Modells sind keine besonderen Kenntnisse über Cedar oder den Dienst Verified Permissions erforderlich. Stattdessen ist es, wie jede andere, in erster Linie eine Übung zur Gestaltung der

Benutzererfahrung und kann sich in Artefakten wie Schnittstellenmodellen, logischen Diagrammen und einer allgemeinen Beschreibung dessen, wie Berechtigungen beeinflussen, was Benutzer im Produkt tun können, äußern. Cedar ist so konzipiert, dass es flexibel genug ist, um Kunden bei einem Modell entgegenzukommen, anstatt das Modell zu zwingen, sich unnatürlich zu verbiegen, um der Implementierung eines Cedar zu entsprechen. Daher ist ein klares Verständnis der gewünschten Benutzererfahrung der beste Weg, um zu einem optimalen Modell zu gelangen.

Gehen Sie wie folgt vor, um die Fragen zu beantworten und zu einem optimalen Modell zu gelangen:

- Weitere Informationen zu den Entwurfsmustern von Cedar finden Sie im Cedar Policy Language Reference Guide.
- Beachten Sie die bewährten Verfahren im Cedar Policy Language Reference Guide.
- Beachten Sie die auf dieser Seite enthaltenen Best Practices.

Bewährte Methoden

- Es gibt kein kanonisches „richtiges“ Modell
- Gib 403 verbotene Fehler statt 404-Fehler „Nicht gefunden“ zurück
- Konzentrieren Sie sich auf Ihre Ressourcen, die über den API-Betrieb hinausgehen
- Überlegungen zur Mehrmandantenfähigkeit

Es gibt kein kanonisches „richtiges“ Modell

Wenn Sie ein Autorisierungsmodell entwerfen, gibt es keine einzige, eindeutig richtige Antwort. Verschiedene Anwendungen können effektiv unterschiedliche Autorisierungsmodelle für ähnliche Konzepte verwenden, und das ist in Ordnung. Stellen Sie sich zum Beispiel die Darstellung des Dateisystems eines Computers vor. Wenn Sie eine Datei in einem UNIX-ähnlichen Betriebssystem erstellen, erbt sie nicht automatisch die Berechtigungen des übergeordneten Ordners. Im Gegensatz dazu erben Dateien in vielen anderen Betriebssystemen und den meisten Online-Filesharing-Diensten die Berechtigungen des übergeordneten Ordners. Beide Optionen sind gültig, je nachdem, für welche Umstände die Anwendung optimiert wird.

Die Richtigkeit einer Autorisierungslösung ist nicht absolut, sondern sollte im Hinblick darauf betrachtet werden, wie sie das von Ihren Kunden gewünschte Erlebnis bietet und ob sie ihre Ressourcen so schützt, wie sie es erwarten. Wenn Ihr Autorisierungsmodell dies erfüllt, ist es erfolgreich.

Aus diesem Grund ist es die hilfreichste Voraussetzung für die Erstellung eines effektiven Autorisierungsmodells, Ihr Design mit der gewünschten Benutzererfahrung zu beginnen.

Gib 403 verbotene Fehler statt 404-Fehler „Nicht gefunden“ zurück

Es empfiehlt sich, bei Anfragen, die eine Entität, insbesondere eine Ressource, enthalten, die keiner Richtlinie entspricht, den Fehler 403 Forbidden zurückzugeben und nicht den Fehler 404 Not found. Dies bietet die höchste Sicherheitsstufe, da Sie nicht offenlegen, ob eine Entität existiert oder nicht, sondern lediglich, dass die Anfrage die Richtlinienbedingungen in keiner Richtlinie im Richtlinienspeicher erfüllt hat.

Konzentrieren Sie sich auf Ihre Ressourcen, die über den API-Betrieb hinausgehen

In den meisten Anwendungen orientieren sich die Berechtigungen an den unterstützten Ressourcen. Beispielsweise kann eine Filesharing-Anwendung Berechtigungen als Aktionen darstellen, die für eine Datei oder einen Ordner ausgeführt werden können. Dies ist ein gutes, einfaches Modell, das die zugrunde liegende Implementierung und die Backend-API-Operationen abstrahiert.

Im Gegensatz dazu entwerfen andere Arten von Anwendungen, insbesondere Webdienste, häufig Berechtigungen rund um die API-Operationen selbst. Wenn ein Webdienst beispielsweise eine API mit dem Namen bereitstelltcreateThing(), könnte das Autorisierungsmodell eine entsprechende Berechtigung definieren, oder eine action in Cedar benanntecreateThing. Dies funktioniert in vielen Situationen und macht es einfach, die Berechtigungen zu verstehen. Um den createThing Vorgang aufzurufen, benötigen Sie die createThing Aktionsberechtigung. Scheint einfach, oder?

Sie werden feststellen, dass der Prozess „Erste Schritte“ in der Verified Permissions-Konsole die Option beinhaltet, Ihre Ressourcen und Aktionen direkt über eine API zu erstellen. Dies ist eine nützliche Grundlage: eine direkte Zuordnung zwischen Ihrem Richtlinienspeicher und der API, für die er autorisiert.

Bei der Weiterentwicklung Ihres Modells ist dieser API-orientierte Ansatz jedoch möglicherweise nicht für Anwendungen mit sehr detaillierten Autorisierungsmodellen geeignet, da sie lediglich ein Proxy für das APIs sind, was Ihre Kunden wirklich zu schützen versuchen: die zugrunde liegenden Daten und Ressourcen. Wenn mehrere Personen den Zugriff auf dieselben Ressourcen APIs kontrollieren, kann es für Administratoren schwierig sein, sich Gedanken über die Pfade zu diesen Ressourcen zu machen und den Zugriff entsprechend zu verwalten.

Stellen Sie sich zum Beispiel ein Benutzerverzeichnis vor, das die Mitglieder einer Organisation enthält. Benutzer können in Gruppen organisiert werden, und eines der Sicherheitsziele besteht darin, die Entdeckung von Gruppenmitgliedschaften durch Unbefugte zu verhindern. Der Dienst, der dieses Benutzerverzeichnis verwaltet, bietet zwei API-Operationen:

- `listMembersOfGroup`
- `listGroupMembershipsForUser`

Kunden können jeden dieser Vorgänge verwenden, um die Gruppenmitgliedschaft zu ermitteln. Daher muss der Berechtigungsadministrator daran denken, den Zugriff auf beide Vorgänge zu koordinieren. Dies wird noch komplizierter, wenn Sie sich später dafür entscheiden, einen neuen API-Vorgang hinzuzufügen, um zusätzliche Anwendungsfälle zu adressieren, wie z. B. die folgenden.

- `isUserInGroups`(eine neue API, um schnell zu testen, ob ein Benutzer zu einer oder mehreren Gruppen gehört)

Aus Sicherheitsgründen eröffnet diese API einen dritten Weg zur Erkennung von Gruppenmitgliedschaften, wodurch die sorgfältig ausgearbeiteten Berechtigungen des Administrators beeinträchtigt werden.

Wir empfehlen, dass Sie sich auf die zugrunde liegenden Daten und Ressourcen und deren Zuordnungsvorgänge konzentrieren. Die Anwendung dieses Ansatzes auf das Beispiel der Gruppenmitgliedschaft würde zu einer abstrakten Berechtigung führen, wie z. B.`viewGroupMembership`, dass für jede der drei API-Operationen eine Abfrage erforderlich ist.

API-Name	Berechtigungen
`listMembersOfGroup`	erfordert eine `viewGroupMembership` Genehmigung für die Gruppe
`listGroupMembershipsForUser`	erfordert `viewGroupMembership` die Erlaubnis des Benutzers
`isUserInGroups`	erfordert `viewGroupMembership` die Erlaubnis des Benutzers

Durch die Definition dieser einen Berechtigung kontrolliert der Administrator erfolgreich den Zugriff auf die Erkennung von Gruppenmitgliedschaften — jetzt und für immer. Als Kompromiss muss nun jeder API-Vorgang die möglicherweise mehreren erforderlichen Berechtigungen dokumentieren, und der Administrator muss diese Dokumentation bei der Erstellung von Berechtigungen heranziehen. Dies kann ein gültiger Kompromiss sein, wenn es notwendig ist, um Ihre Sicherheitsanforderungen zu erfüllen.

Überlegungen zur Mehrmandantenfähigkeit

Möglicherweise möchten Sie Anwendungen für die Nutzung durch mehrere Kunden — Unternehmen, die Ihre Anwendung nutzen, oder Mandanten — entwickeln und sie in Amazon Verified Permissions integrieren. Bevor Sie Ihr Autorisierungsmodell entwickeln, sollten Sie eine Mehrmandantenstrategie entwickeln. Sie können die Richtlinien Ihrer Kunden in einem gemeinsamen Richtlinienspeicher verwalten oder jedem einzelnen Mandanten einen Richtlinienspeicher zuweisen. Weitere Informationen finden Sie unter Überlegungen zum Multi-Tenant-Design von Amazon Verified Permissions in AWS Prescriptive Guidance.

1. Ein gemeinsam genutzter Richtlinienspeicher

   Alle Mandanten teilen sich einen einzigen Richtlinienspeicher. Die Anwendung sendet alle Autorisierungsanfragen an den gemeinsamen Richtlinienspeicher.

2. Richtlinienspeicher pro Mandant

   Jeder Mandant hat einen eigenen Richtlinienspeicher. Die Anwendung fragt je nach Mandant, der die Anfrage stellt, verschiedene Richtlinienspeicher nach einer Autorisierungsentscheidung ab.

Keine der beiden Strategien wird große Auswirkungen auf Ihre AWS Rechnung haben. Wie sollten Sie dann Ihren Ansatz gestalten? Im Folgenden finden Sie allgemeine Bedingungen, die zu Ihrer Mehrmandanten-Autorisierungsstrategie von Verified Permissions beitragen könnten.

Isolierung der Mandantenrichtlinien

Die Isolierung der Richtlinien der einzelnen Mandanten von den anderen ist wichtig, um die Mandantendaten zu schützen. Wenn jeder Mandant seinen eigenen Richtlinienspeicher hat, hat jeder seine eigenen isolierten Richtlinien.

Ablauf der Autorisierung

Sie können einen Mandanten, der eine Autorisierungsanfrage stellt, anhand einer Policy-Store-ID in der Anfrage identifizieren, und zwar mit Policy-Stores pro Mandant. Bei einem gemeinsam genutzten Richtlinienspeicher verwenden alle Anfragen dieselbe Richtlinienspeicher-ID.

Vorlagen und Schemaverwaltung

Wenn Ihre Anwendung über mehrere Richtlinienspeicher verfügt, erhöhen Ihre Richtlinienvorlagen und ein Richtlinienspeicherschema den Entwurfs- und Wartungsaufwand für jeden Richtlinienspeicher.

Globales Richtlinienmanagement

Möglicherweise möchten Sie einige globale Richtlinien auf jeden Mandanten anwenden. Die Höhe des Verwaltungsaufwands für globale Richtlinien variiert je nach Modell mit gemeinsam genutztem und mandantenspezifischem Policy-Store.

Ausgliederung von Mandanten

Einige Mieter werden Elemente zu Ihrem Schema und Ihren Richtlinien beitragen, die für ihren Fall spezifisch sind. Wenn ein Mandant nicht mehr in Ihrer Organisation aktiv ist und Sie seine Daten entfernen möchten, hängt der Aufwand davon ab, inwieweit er von anderen Mandanten isoliert ist.

Kontingente für Service-Ressourcen

Verified Permissions verfügt über Ressourcen- und Anforderungsquoten, die Ihre Entscheidung für mehrere Mandanten beeinflussen können. Weitere Informationen zu Kontingenten finden Sie unter Kontingente für Ressourcen.

Vergleich von gemeinsam genutzten Richtlinienspeichern und Richtlinienspeichern pro Mandant

Jede Überlegung erfordert ihr eigenes Maß an Zeit und Ressourcen in Form von gemeinsam genutzten und mandantenspezifischen Policy-Store-Modellen.

Überlegungen	Umfang des Aufwands in einem gemeinsamen Richtlinienspeicher	Umfang des Aufwands in den Policy-Speichern pro Mandant

Isolierung von Mandanten richtlinien	Mittel.Muss Mandanten kennungen in Richtlinien und Autorisierungsanfragen enthalten.	Niedrig. Isolation ist das Standardverhalten. Auf mandantenspezifische Richtlinien können andere Mandanten nicht zugreifen.
Ablauf der Autorisierung	Niedrig. Alle Abfragen zielen auf einen Richtlinienspeicher ab.	Mittel. Muss die Zuordnungen zwischen jedem Mandanten und seiner Policy-Store-ID beibehalten.
Vorlagen und Schemaver waltung	Niedrig. Muss dafür sorgen, dass ein Schema für alle Mandanten funktioniert.	Hoch. Schemas und Vorlagen mögen für sich genommen weniger komplex sein, Änderungen erfordern jedoch mehr Koordination und Komplexität.
Globales Policy-Management	Niedrig. Alle Richtlinien sind global und können zentral aktualisiert werden.	Hoch. Beim Onboarding müssen Sie jedem Richtlini enspeicher globale Richtlini en hinzufügen. Replizieren Sie globale Richtlinienaktuali sierungen zwischen vielen Richtlinienspeichern.
Ausgliederung von Mietern	Hoch. Es müssen nur mandantenspezifische Richtlinien identifiziert und gelöscht werden.	Niedrig. Löschen Sie den Richtlinienspeicher.

Kontingente für Service-Ressourcen	Hoch. Mandanten teilen sich Ressourcenkontingente, die sich auf Richtlinienspeicher wie Schemagröße, Richtlinengröße pro Ressource und Identitätsquellen pro Richtlinienspeicher auswirken.	Niedrig. Jeder Mandant hat eigene Ressourcenkontingente.

Wie soll man wählen

Jede Multi-Tenant-Anwendung ist anders. Vergleichen Sie die beiden Ansätze und ihre Überlegungen sorgfältig, bevor Sie eine architektonische Entscheidung treffen.

Wenn Ihre Anwendung keine mandantenspezifischen Richtlinien erfordert und eine einzige Identitätsquelle verwendet, ist ein gemeinsamer Richtlinienspeicher für alle Mandanten wahrscheinlich die effektivste Lösung. Dies führt zu einem einfacheren Autorisierungsablauf und einer globalen Richtlinienverwaltung. Das Offboarding eines Mandanten mithilfe eines gemeinsamen Richtlinienspeichers erfordert weniger Aufwand, da die Anwendung keine mandantenspezifischen Richtlinien löschen muss.

Wenn Ihre Anwendung jedoch viele mandantenspezifische Richtlinien erfordert oder mehrere Identitätsquellen verwendet, sind richtlinienspeicher pro Mandant wahrscheinlich am effektivsten. Sie können den Zugriff auf Mandantenrichtlinien mit IAM Richtlinien steuern, die jedem Richtlinienspeicher mandantenspezifische Berechtigungen gewähren. Das Offboarding eines Mandanten beinhaltet das Löschen seines Richtlinienspeichers. In einer shared-policy-store Umgebung müssen Sie mandantenspezifische Richtlinien finden und löschen.

Richtlinien von Amazon Verified Permissions

Ein Richtlinienspeicher ist ein Container für Richtlinien und Richtlinienvorlagen. In jedem Richtlinienspeicher können Sie ein Schema erstellen, das zur Validierung der zum Richtlinienspeicher hinzugefügten Richtlinien verwendet wird. Darüber hinaus können Sie die Richtlinienvalidierung aktivieren. Wenn Sie eine Richtlinie mit aktivierter Richtlinienvalidierung zu einem Richtlinienspeicher hinzufügen, werden die in der Richtlinie definierten Entitätstypen, allgemeinen Typen und Aktionen anhand des Schemas validiert und ungültige Richtlinien werden abgelehnt.

Der Löschschutz verhindert das versehentliche Löschen eines Richtlinienspeichers. Der Löschschutz ist für alle neuen Richtlinienspeicher aktiviert, die über den erstellt wurden AWS Management Console. Im Gegensatz dazu ist er für alle Richtlinienspeicher deaktiviert, die über einen API- oder SDK-Aufruf erstellt wurden.

Wir empfehlen, einen Richtlinienspeicher pro Anwendung oder einen Richtlinienspeicher pro Mandant für Anwendungen mit mehreren Mandanten zu erstellen. Sie müssen einen Richtlinienspeicher angeben, wenn Sie eine Autorisierungsanfrage stellen.

Wir empfehlen, in Ihren Richtlinienspeichern Namespaces für Cedar-Entitäten zu verwenden, um Unklarheiten zu vermeiden. Ein Namespace ist ein Zeichenkettenpräfix für einen Typ, getrennt durch ein Paar Doppelpunkte () als Trennzeichen. `::` Zum Beispiel `MyApplicationNamespace::exampleType`. Verified Permissions unterstützt einen Namespace pro Richtlinienspeicher. Diese Namespaces helfen dabei, die Dinge im Griff zu behalten, wenn Sie mit mehreren ähnlichen Anwendungen arbeiten. Wenn Sie beispielsweise in Mehrmandantenanwendungen einen Namespace verwenden, um den Namen des Mandanten an die im Schema definierten Typen anzuhängen, unterscheiden sich diese von ihren ähnlichen Gegenstücken, die von den anderen Mandanten verwendet werden. Wenn Sie sich die Protokolle der Autorisierungsanfragen ansehen, können Sie leicht den Mandanten identifizieren, der die Autorisierungsanfrage bearbeitet hat. Weitere Informationen finden Sie unter Namespaces im Cedar Policy Language Reference Guide.

Themen

- Richtlinienspeicher für verifizierte Berechtigungen erstellen
- API-verknüpfte Richtlinienspeicher
- Policy-Stores löschen

Richtlinienspeicher für verifizierte Berechtigungen erstellen

Sie können einen Richtlinienspeicher mit den folgenden Methoden erstellen:

- Folgen Sie einer Anleitung zur Einrichtung — Sie definieren einen Ressourcentyp mit gültigen Aktionen und einen Prinzipaltyp, bevor Sie Ihre erste Richtlinie erstellen.
- Mit API Gateway und einer Identitätsquelle einrichten — Definieren Sie Ihre Hauptentitäten mit Benutzern, die sich mit einem Identitätsanbieter (IdP) anmelden, und Ihre Aktionen und Ressourcenentitäten über eine Amazon API Gateway Gateway-API. Wir empfehlen diese Option, wenn Sie möchten, dass Ihre Anwendung API-Anfragen mit der Gruppenmitgliedschaft von Benutzern oder anderen Attributen autorisiert.
- Beginnen Sie mit einem Beispielrichtlinienspeicher — Wählen Sie einen vordefinierten Beispiel-Richtlinienspeicher für Projekte aus. Wir empfehlen diese Option, wenn Sie mehr über verifizierte Berechtigungen erfahren und Beispielrichtlinien ansehen und testen möchten.
- Erstellen Sie einen leeren Richtlinienspeicher — Sie definieren das Schema und alle Zugriffsrichtlinien selbst. Wir empfehlen diese Option, wenn Sie bereits mit der Konfiguration eines Richtlinienspeichers vertraut sind.

Guided setup

So erstellen Sie einen Richtlinienspeicher mithilfe der Konfigurationsmethode „Geführtes Setup“

Der Assistent für die Einrichtung mit Anleitung führt Sie durch den Prozess der Erstellung der ersten Iteration Ihres Richtlinienspeichers. Sie erstellen ein Schema für Ihren ersten Ressourcentyp, beschreiben die Aktionen, die für diesen Ressourcentyp gelten, und beschreiben den Prinzipaltyp, für den Sie Berechtigungen erteilen. Anschließend erstellen Sie Ihre erste Richtlinie. Sobald Sie diesen Assistenten abgeschlossen haben, können Sie Ihrem Richtlinienspeicher etwas hinzufügen, das Schema erweitern, um andere Ressourcen- und Prinzipaltypen zu beschreiben, und zusätzliche Richtlinien und Vorlagen erstellen.

1. Wählen Sie in der Konsole „Verifizierte Berechtigungen“ die Option Neuen Richtlinienspeicher erstellen aus.
2. Wählen Sie im Abschnitt Startoptionen die Option Geführte Installation aus.
3. Geben Sie eine Beschreibung des Policy-Stores ein. Bei diesem Text kann es sich um einen beliebigen Text handeln, der zu Ihrer Organisation passt, und zwar als freundliche Referenz für die Funktion des aktuellen Richtlinienspeichers, z. B. die Webanwendung für Wetteraktualisierungen.

4. Geben Sie im Abschnitt Details einen Namespace für Ihr Schema ein. Weitere Hinweise zu Namespaces finden Sie unter. Namespace-Definition
5. Wählen Sie Weiter.
6. Geben Sie im Fenster Ressourcentyp einen Namen für Ihren Ressourcentyp ein. Dies `currentTemperature` könnte beispielsweise eine Ressource für die Webanwendung für Wetteraktualisierungen sein.
7. (Optional) Wählen Sie Attribut hinzufügen aus, um Ressourcenattribute hinzuzufügen. Geben Sie den Attributnamen ein und wählen Sie einen Attributtyp für jedes Attribut der Ressource aus. Wählen Sie aus, ob jedes Attribut erforderlich ist. `temperatureFormat`Könnte beispielsweise ein Attribut für die `currentTemperature` Ressource sein und entweder Fahrenheit oder Celsius sein. Um ein Attribut zu entfernen, das für den Ressourcentyp hinzugefügt wurde, wählen Sie neben dem Attribut die Option Entfernen aus.
8. Geben Sie im Feld Aktionen die Aktionen ein, die für den angegebenen Ressourcentyp autorisiert werden sollen. Um weitere Aktionen für den Ressourcentyp hinzuzufügen, wählen Sie Aktion hinzufügen aus. Dies `viewTemperature` könnte beispielsweise eine Aktion in der Webanwendung für Wetteraktualisierungen sein. Um eine Aktion zu entfernen, die für den Ressourcentyp hinzugefügt wurde, wählen Sie neben der Aktion die Option Entfernen aus.
9. Geben Sie im Feld Name des Prinzipaltyps den Namen für einen Prinzipaltyp ein, der die angegebenen Aktionen für Ihren Ressourcentyp verwenden wird. Standardmäßig wird Benutzer zu diesem Feld hinzugefügt, kann aber ersetzt werden.
10. Wählen Sie Weiter.
11. Wählen Sie im Fenster Prinzipaltyp die Identitätsquelle für Ihren Prinzipaltyp aus.

    - Wählen Sie Benutzerdefiniert, wenn die ID und die Attribute des Prinzipals direkt von Ihrer Verified Permissions-Anwendung bereitgestellt werden. Wählen Sie Attribut hinzufügen aus, um Hauptattribute hinzuzufügen. Verified Permissions verwendet die angegebenen Attributwerte, wenn Richtlinien anhand des Schemas überprüft werden. Um ein Attribut zu entfernen, das für den Prinzipaltyp hinzugefügt wurde, wählen Sie neben dem Attribut die Option Entfernen aus.
    - Wählen Sie Cognito User Pool, wenn die ID und die Attribute des Prinzipals aus einer von Amazon Cognito generierten ID oder einem Zugriffstoken bereitgestellt werden. Wählen Sie Connect user pool. Wählen Sie die AWS-RegionBenutzerpool-ID des Amazon Cognito Cognito-Benutzerpools aus, zu dem Sie eine Verbindung herstellen möchten, und geben Sie sie ein. Wählen Sie Connect aus. Weitere Informationen finden Sie unter Autorisierung mit von Amazon verifizierten Berechtigungen im Amazon Cognito Developer Guide.

- Wählen Sie Externer OIDC-Anbieter, wenn die ID und die Attribute des Prinzipals aus einem ID- und/oder Zugriffstoken extrahiert werden, das von einem externen OIDC-Anbieter generiert wurde, und fügen Sie die Anbieter- und Token-Details hinzu.

12. Wählen Sie Weiter.
13. Geben Sie im Abschnitt Richtliniendetails eine optionale Richtlinienbeschreibung für Ihre erste Cedar-Richtlinie ein.
14. Wählen Sie im Feld Principals Scope die Principals aus, denen im Rahmen der Richtlinie Berechtigungen erteilt werden sollen.

    - Wählen Sie Spezifischer Hauptbenutzer aus, um die Richtlinie auf einen bestimmten Prinzipal anzuwenden. Wählen Sie den Principal im Feld Principal, der Aktionen ausführen darf, und geben Sie eine Entitäts-ID für den Principal ein. Dies `user-id` könnte beispielsweise eine Entitäts-ID in der Webanwendung für Wetteraktualisierungen sein.

      > **Note**
      >
      > Wenn Sie Amazon Cognito verwenden, muss die Entitäts-ID als formatiert sein. `<userpool-id>|<sub>`

    - Wählen Sie Alle Prinzipale, um die Richtlinie auf alle Prinzipale in Ihrem Richtlinienspeicher anzuwenden.

15. Wählen Sie im Feld Umfang der Ressourcen aus, auf welche Ressourcen die angegebenen Prinzipale reagieren dürfen.

    - Wählen Sie Bestimmte Ressource aus, um die Richtlinie auf eine bestimmte Ressource anzuwenden. Wählen Sie die Ressource im Feld Ressource, für die diese Richtlinie gelten soll, und geben Sie eine Entitäts-ID für die Ressource ein. Dies `temperature-id` könnte beispielsweise eine Entitäts-ID in der Webanwendung für Wetteraktualisierungen sein.
    - Wählen Sie Alle Ressourcen aus, um die Richtlinie auf alle Ressourcen in Ihrem Richtlinienspeicher anzuwenden.

16. Wählen Sie im Feld Aktionsbereich aus, für welche Aktionen die angegebenen Prinzipale autorisiert werden sollen.

    - Wählen Sie Spezifische Gruppe von Aktionen aus, um die Richtlinie auf bestimmte Aktionen anzuwenden. Aktivieren Sie die Kontrollkästchen neben den Aktionen im Feld Aktion (en), für die diese Richtlinie gelten soll.

- Wählen Sie Alle Aktionen aus, um die Richtlinie auf alle Aktionen in Ihrem Richtlinienspeicher anzuwenden.

17. Sehen Sie sich die Richtlinie im Abschnitt Richtlinienvorschau an. Wählen Sie Richtlinienspeicher erstellen aus.

Set up with API Gateway and an identity source

So erstellen Sie einen Richtlinienspeicher mithilfe der Konfigurationsmethode „Mit API Gateway einrichten“ und einer Identitätsquelle

Die API-Gateway-Option schützt APIs mit Richtlinien für verifizierte Berechtigungen, die darauf ausgelegt sind, Autorisierungsentscheidungen anhand von Benutzergruppen oder Rollen zu treffen. Diese Option erstellt einen Richtlinienspeicher zum Testen der Autorisierung mit Identitätsquellengruppen und eine API mit einem Lambda-Autorisierer.

Die Benutzer und ihre Gruppen in einem IdP werden entweder zu Ihren Prinzipalen (ID-Token) oder zu Ihrem Kontext (Zugriffstoken). Die Methoden und Pfade in einer API-Gateway-API werden zu den Aktionen, die Ihre Richtlinien autorisieren. Ihre Anwendung wird zur Ressource. Als Ergebnis dieses Workflows erstellt Verified Permissions einen Richtlinienspeicher, eine Lambda-Funktion und einen API-Lambda-Authorizer. Sie müssen den Lambda-Autorisierer Ihrer API zuweisen, nachdem Sie diesen Workflow abgeschlossen haben.

1. Wählen Sie in der Konsole „Verifizierte Berechtigungen“ die Option Neuen Richtlinienspeicher erstellen aus.
2. Wählen Sie im Abschnitt Startoptionen die Option Mit API Gateway und einer Identitätsquelle einrichten und dann Weiter aus.
3. Wählen Sie im Schritt Ressourcen und Aktionen importieren unter API eine API aus, die als Modell für die Ressourcen und Aktionen Ihres Richtlinienspeichers dienen soll.

    a. Wählen Sie aus den in Ihrer API konfigurierten Phasen eine Bereitstellungsphase aus und wählen Sie API importieren aus. Weitere Informationen zu API-Phasen finden Sie im Amazon API Gateway Developer Guide unter Setting up a Stage for a REST API API API API API API API API.

    b. Sehen Sie sich eine Vorschau Ihrer Map mit importierten Ressourcen und Aktionen an.

    c. Um Ressourcen oder Aktionen zu aktualisieren, ändern Sie Ihre API-Pfade oder Methoden in der API Gateway Gateway-Konsole und wählen Sie API importieren aus, um die Updates zu sehen.

d. Wenn Sie mit Ihrer Auswahl zufrieden sind, wählen Sie Weiter.

4. Wählen Sie unter Identitätsquelle einen Identitätsanbietertyp aus. Sie können einen Amazon Cognito Cognito-Benutzerpool oder einen OpenID Connect (OIDC) IdP-Typ wählen.
5. Wenn Sie sich für Amazon Cognito entschieden haben:

   a. Wählen Sie einen Benutzerpool aus, der sich in demselben AWS-Region und AWS-Konto wie in Ihrem Richtlinienspeicher befindet.

   b. Wählen Sie den Tokentyp aus, der an die API übergeben werden soll und den Sie zur Autorisierung einreichen möchten. Beide Tokentypen enthalten Benutzergruppen, die Grundlage dieses API-verknüpften Autorisierungsmodells.

   c. Unter App-Client-Validierung können Sie den Umfang eines Policy Stores auf eine Teilmenge der Amazon Cognito-App-Clients in einem Benutzerpool mit mehreren Mandanten beschränken. Um zu verlangen, dass sich dieser Benutzer bei einem oder mehreren angegebenen App-Clients in Ihrem Benutzerpool authentifiziert, wählen Sie Nur Tokens mit erwartetem App-Client akzeptieren aus. IDs Um jeden Benutzer zu akzeptieren, der sich beim Benutzerpool authentifiziert, wählen Sie App-Client nicht validieren aus. IDs

   d. Wählen Sie Weiter.

6. Wenn Sie sich für einen externen OIDC-Anbieter entschieden haben:

   a. Geben Sie im Feld Aussteller-URL die URL Ihres OIDC-Ausstellers ein. Dies ist der Dienstendpunkt, der beispielsweise den Autorisierungsserver, Signaturschlüssel und andere Informationen über Ihren Anbieter bereitstellt. `https://auth.example.com` Ihre Aussteller-URL muss ein OIDC-Discovery-Dokument unter hosten. `/.well-known/openid-configuration`

   b. Wählen Sie unter Tokentyp den Typ des OIDC JWT aus, den Ihre Anwendung zur Autorisierung einreichen soll. Weitere Informationen finden Sie unter Zuordnen von Identitätsanbieter-Tokens zum Schema.

   c. (optional) Wählen Sie unter Token-Ansprüche — optional die Option Token-Anspruch hinzufügen aus, geben Sie einen Namen für das Token ein und wählen Sie einen Wertetyp aus.

   d. Gehen Sie unter Benutzer- und Gruppentoken-Ansprüche wie folgt vor:

      i. Geben Sie im Feld Token einen Namen für den Benutzeranspruch für die Identitätsquelle ein. Dabei handelt es sich in der Regel `sub` um einen Anspruch aus

Ihrer ID oder Ihrem Zugriffstoken, das die eindeutige Kennung für die zu prüfende Entität enthält. Identitäten des verbundenen OIDC-IdP werden dem Benutzertyp in Ihrem Richtlinienspeicher zugeordnet.

ii. Geben Sie einen Namen für den Gruppenanspruch als Token für die Identitätsquelle ein. Dabei handelt es sich in der Regel `groups` um einen Anspruch aus Ihrer ID oder Ihrem Zugriffstoken, der eine Liste der Benutzergruppen enthält. Ihr Richtlinienspeicher autorisiert Anfragen auf der Grundlage der Gruppenmitgliedschaft.

e. Wählen Sie unter Zielgruppenvalidierung einen Wert aus, den Ihr Richtlinienspeicher in Autorisierungsanfragen akzeptieren soll, `Add value` und fügen Sie ihn hinzu.

f. Wählen Sie Weiter.

7. Wenn Sie sich für Amazon Cognito entschieden haben, fragt Verified Permissions Ihren Benutzerpool nach Gruppen ab. Geben Sie für OIDC-Anbieter Gruppennamen manuell ein. Mit dem Schritt Aktionen Gruppen zuweisen werden Richtlinien für Ihren Richtlinienspeicher erstellt, die es Gruppenmitgliedern ermöglichen, Aktionen auszuführen.

   a. Wählen Sie die Gruppen aus, die Sie in Ihre Richtlinien aufnehmen möchten, oder fügen Sie sie hinzu.

   b. Weisen Sie jeder der ausgewählten Gruppen Aktionen zu.

   c. Wählen Sie Weiter.

8. Wählen Sie unter Anwendungsintegration bereitstellen aus, ob Sie den Lambda-Autorisierer später manuell anhängen möchten oder ob Verified Permissions dies jetzt für Sie erledigen soll, und überprüfen Sie die Schritte, die Verified Permissions zur Erstellung Ihres Richtlinienspeichers und Lambda-Autorisierers unternimmt.

9. Wenn Sie bereit sind, die neuen Ressourcen zu erstellen, wählen Sie Create Policy Store aus.

10. Lassen Sie den Schritt Status des Richtlinienspeichers in Ihrem Browser geöffnet, um den Fortschritt der Ressourcenerstellung anhand von Verified Permissions zu überwachen.

11. Wenn Sie sich entschieden haben, den Autorisierer manuell anzuhängen, konfigurieren Sie Ihren Autorisierer nach einiger Zeit, normalerweise etwa einer Stunde, oder wenn der Schritt Lambda-Authorizer bereitstellen Success anzeigt.

    Verified Permissions hat eine Lambda-Funktion und einen Lambda-Authorizer in Ihrer API erstellt. Wählen Sie Open API, um zu Ihrer API zu navigieren.

Informationen zum Zuweisen eines Lambda-Autorisierers finden Sie unter Verwenden von API Gateway Gateway-Lambda-Autorisierern im Amazon API Gateway Gateway-Entwicklerhandbuch.

a. Navigieren Sie zu Autorisatoren für Ihre API und notieren Sie sich den Namen des Autorisierers, den Verified Permissions erstellt hat.
b. Navigieren Sie zu Ressourcen und wählen Sie eine Methode der obersten Ebene in Ihrer API aus.
c. Wählen Sie unter Einstellungen für Methodenanfragen die Option Bearbeiten aus.
d. Geben Sie als Autorisierer den Namen des Autorisierers ein, den Sie sich zuvor notiert haben.
e. Erweitern Sie HTTP-Anforderungsheader, geben Sie einen Namen oder ein und wählen Sie **`AUTHORIZATION`** Erforderlich aus.
f. Stellen Sie die API-Phase bereit.
g. Speichern Sie Ihre Änderungen.

12. Testen Sie Ihren Autorisierer mit einem Benutzerpool-Token des Tokentyps, den Sie im Schritt Identitätsquelle auswählen ausgewählt haben. Weitere Informationen zur Benutzerpool-Anmeldung und zum Abrufen von Token finden Sie unter Ablauf der Benutzerpool-Authentifizierung im Amazon Cognito Developer Guide.
13. Testen Sie die Authentifizierung erneut mit einem Benutzerpool-Token im `AUTHORIZATION` Header einer Anfrage an Ihre API.
14. Untersuchen Sie Ihren neuen Richtlinienspeicher. Fügen Sie Richtlinien hinzu und verfeinern Sie sie.

Sample policy store

So erstellen Sie einen Richtlinienspeicher mit der Beispiel-Konfigurationsmethode für den Richtlinienspeicher

1. Wählen Sie im Abschnitt Startoptionen die Option Beispiel für einen Richtlinienspeicher aus.
2. Wählen Sie im Abschnitt Beispielprojekt den Typ der Beispielanwendung Verified Permissions aus, die Sie verwenden möchten.

   - PhotoFlashist eine Beispiel-Webanwendung für Kunden, mit der Benutzer einzelne Fotos und Alben mit Freunden teilen können. Benutzer können detaillierte Berechtigungen dafür

festlegen, wer ihre Fotos ansehen, kommentieren und erneut teilen darf. Kontoinhaber können auch Gruppen von Freunden erstellen und Fotos in Alben organisieren.

- DigitalPetStoreist eine Beispielanwendung, bei der sich jeder registrieren und Kunde werden kann. Kunden können Haustiere zum Verkauf hinzufügen, nach Haustieren suchen und Bestellungen aufgeben. Kunden, die ein Haustier hinzugefügt haben, werden als Tierbesitzer registriert. Tierbesitzer können die Angaben zum Haustier aktualisieren, ein Tierbild hochladen oder den Tiereintrag löschen. Kunden, die eine Bestellung aufgegeben haben, werden als Inhaber der Bestellung registriert. Bestellungsinhaber können Einzelheiten zur Bestellung abrufen oder sie stornieren. Manager von Tierhandlungen haben Administratorzugriff.

  > **Note**
  >
  > Der DigitalPetStoreBeispiel-Richtlinienspeicher enthält keine Richtlinienvorlagen. Der Richtlinienspeicher PhotoFlashund der TinyTodoBeispielrichtlinienspeicher enthalten Richtlinienvorlagen.

- TinyTodoist eine Beispielanwendung, mit der Benutzer Aufgaben und Aufgabenlisten erstellen können. Listenbesitzer können ihre Listen verwalten und teilen und angeben, wer ihre Listen ansehen oder bearbeiten kann.

3. Basierend auf dem ausgewählten Beispielprojekt wird automatisch ein Namespace für das Schema Ihres Beispielrichtlinienspeichers generiert.
4. Wählen Sie Richtlinienspeicher erstellen aus.

   Ihr Richtlinienspeicher wird mit Richtlinien und einem Schema für den von Ihnen ausgewählten Beispiel-Richtlinienspeicher erstellt. Weitere Informationen zu vorlagenverknüpften Richtlinien, die Sie für die Beispielrichtlinienspeicher erstellen können, finden Sie unter. Beispiel für vorlagenverknüpfte Richtlinien mit Amazon Verified Permissions

Empty policy store

So erstellen Sie einen Richtlinienspeicher mit der Konfigurationsmethode „Richtlinienspeicher leeren“

1. Wählen Sie im Abschnitt Startoptionen die Option Richtlinienspeicher leeren aus.
2. Wählen Sie Richtlinienspeicher erstellen aus.

Ein leerer Richtlinienspeicher wird ohne Schema erstellt, was bedeutet, dass Richtlinien nicht validiert werden. Weitere Informationen zur Aktualisierung des Schemas für Ihren Richtlinienspeicher finden Sie unterSpeicherschema für Richtlinien von Amazon Verified Permissions.

Weitere Informationen zum Erstellen von Richtlinien für Ihren Richtlinienspeicher finden Sie unter Statische Richtlinien für Amazon Verified Permissions erstellen undMit Vorlagen verknüpfte Richtlinien mit Amazon Verified Permissions erstellen.

AWS CLI

Um einen leeren Richtlinienspeicher zu erstellen, verwenden Sie den AWS CLI.

Sie können einen Richtlinienspeicher erstellen, indem Sie den `create-policy-store` Vorgang verwenden.

> **Note**
>
> Ein Richtlinienspeicher, den Sie mithilfe von erstellen, AWS CLI ist leer.
>
> - Informationen zum Hinzufügen eines Schemas finden Sie unterSpeicherschema für Richtlinien von Amazon Verified Permissions.
> - Informationen zum Hinzufügen von Richtlinien finden Sie unterStatische Richtlinien für Amazon Verified Permissions erstellen.
> - Informationen zum Hinzufügen von Richtlinienvorlagen finden Sie unterRichtlinienvorlagen für Amazon Verified Permissions erstellen.

```
$ aws verifiedpermissions create-policy-store \
    --validation-settings "mode=STRICT"
{
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
    "createdDate": "2023-05-16T17:41:29.103459+00:00",
    "lastUpdatedDate": "2023-05-16T17:41:29.103459+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

AWS SDKs

Sie können mithilfe der `CreatePolicyStore` API einen Richtlinienspeicher erstellen. Weitere Informationen finden Sie CreatePolicyStoreim Referenzhandbuch zur Amazon Verified Permissions API.

Implementierung von Amazon Verified Permissions in Rust mit dem AWS SDK

Dieses Thema bietet ein praktisches Beispiel für die Implementierung von Amazon Verified Permissions in Rust mit dem AWS SDK. Dieses Beispiel zeigt, wie ein Autorisierungsmodell entwickelt wird, mit dem getestet werden kann, ob ein Benutzer ein Foto ansehen kann. Der Beispielcode verwendet die aws-sdk-verifiedpermissionsCrate aus dem AWS SDK für Rust, das eine Reihe robuster Tools für die Interaktion mit AWS Diensten bietet.

Voraussetzungen

Stellen Sie vor dem Start sicher, dass Sie die AWS CLI auf Ihrem System konfiguriert haben und dass Sie mit Rust vertraut sind.

- Anweisungen zur Installation von finden Sie in der AWS CLIAWS CLI-Installationsanleitung.
- Anweisungen zur Konfiguration von finden Sie unter Konfiguration der AWS CLI Einstellungen für und Einstellungen für Konfiguration AWS CLI und Anmeldeinformationsdatei in der AWS CLI.
- Weitere Informationen zu rust finden Sie auf rust-lang.org und im AWS SDK for Rust Developer Guide.

Nachdem Ihre Umgebung vorbereitet ist, wollen wir untersuchen, wie verifizierte Berechtigungen in Rust implementiert werden können.

Testen Sie den Beispielcode

Der Beispielcode macht Folgendes:

- Richtet den SDK-Client für die Kommunikation ein AWS
- Erstellt einen Richtlinienspeicher
- Definiert die Struktur des Richtlinienspeichers durch Hinzufügen eines Schemas

- Fügt eine Richtlinie zur Überprüfung von Autorisierungsanfragen hinzu
- Sendet eine Testautorisierungsanfrage, um zu überprüfen, ob alles korrekt eingerichtet ist

So testen Sie den Beispiel-Code

1. Erstellen Sie ein Rust-Projekt.
2. Ersetzen Sie jeden vorhandenen Code `main.rs` durch den folgenden Code:

```
use std::time::Duration;
use std::thread::sleep;
use aws_config::BehaviorVersion;
use aws_sdk_verifiedpermissions::Client;
use aws_sdk_verifiedpermissions::{
    operation::{
        create_policy::CreatePolicyOutput,
        create_policy_store::CreatePolicyStoreOutput,
        is_authorized::IsAuthorizedOutput,
        put_schema::PutSchemaOutput,
    },
    types::{
        ActionIdentifier, EntityIdentifier, PolicyDefinition, SchemaDefinition,
 StaticPolicyDefinition, ValidationSettings
    },
};

//Function that creates a policy store in the client that's passed
async fn create_policy_store(client: &Client, valid_settings: &ValidationSettings)-
> CreatePolicyStoreOutput {
    let policy_store =
 client.create_policy_store().validation_settings(valid_settings.clone()).send().await;
    return policy_store.unwrap();
}

//Function that adds a schema to the policy store in the client
async fn put_schema(client: &Client, ps_id: &str, schema: &str) -> PutSchemaOutput
 {
    let schema =
 client.put_schema().definition(SchemaDefinition::CedarJson(schema.to_string())).policy_sto
    return schema.unwrap();
}

//Function that creates a policy in the policy store in the client
```

```
async fn create_policy(client: &Client, ps_id: &str,
 policy_definition:&PolicyDefinition) -> CreatePolicyOutput {
    let create_policy =
 client.create_policy().definition(policy_definition.clone()).policy_store_id(ps_id).send()
    return create_policy.unwrap();
}

//Function that tests the authorization request to the policy store in the client
async fn authorize(client: &Client, ps_id: &str, principal: &EntityIdentifier,
 action: &ActionIdentifier, resource: &EntityIdentifier) -> IsAuthorizedOutput {
    let is_auth =
 client.is_authorized().principal(principal.to_owned()).action(action.to_owned()).resource(
    return is_auth.unwrap();
}

#[::tokio::main]
async fn main() -> Result<(), aws_sdk_verifiedpermissions::Error> {

//Set up SDK client
    let config = aws_config::load_defaults(BehaviorVersion::latest()).await;
    let client = aws_sdk_verifiedpermissions::Client::new(&config);

//Create a policy store
    let valid_settings = ValidationSettings::builder()
    .mode({aws_sdk_verifiedpermissions::types::ValidationMode::Strict
    })
    .build()
    .unwrap();
    let policy_store = create_policy_store(&client, &valid_settings).await;
    println!(
    "Created Policy store with ID: {:?}",
    policy_store.policy_store_id
    );

//Add schema to policy store
    let schema= r#"{
        "PhotoFlash": {
            "actions": {
                "ViewPhoto": {
                    "appliesTo": {
                        "context": {
                            "type": "Record",
                            "attributes": {}
                        },
```

```
                        "principalTypes": [
                            "User"
                        ],
                        "resourceTypes": [
                            "Photo"
                        ]
                    },
                    "memberOf": []
                }
            },
            "entityTypes": {
                "Photo": {
                    "memberOfTypes": [],
                    "shape": {
                        "type": "Record",
                        "attributes": {
                            "IsPrivate": {
                                "type": "Boolean"
                            }
                        }
                    }
                },
                "User": {
                    "memberOfTypes": [],
                    "shape": {
                        "attributes": {},
                        "type": "Record"
                    }
                }
            }
        }
    }"#;
    let put_schema = put_schema(&client, &policy_store.policy_store_id,
 schema).await;
    println!(
        "Created Schema with Namespace: {:?}",
        put_schema.namespaces
    );

//Create policy
    let policy_text = r#"
        permit (
            principal in PhotoFlash::User::"alice",
            action == PhotoFlash::Action::"ViewPhoto",
```

```
            resource == PhotoFlash::Photo::"VacationPhoto94.jpg"
        );
        "#;
    let policy_definition =
 PolicyDefinition::Static(StaticPolicyDefinition::builder().statement(policy_text).build().
    let policy = create_policy(&client, &policy_store.policy_store_id,
 &policy_definition).await;
    println!(
        "Created Policy with ID: {:?}",
        policy.policy_id
    );

//Break to make sure the resources are created before testing authorization
    sleep(Duration::new(2, 0));

//Test authorization
    let principal=
 EntityIdentifier::builder().entity_id("alice").entity_type("PhotoFlash::User").build().unw
    let action =
 ActionIdentifier::builder().action_type("PhotoFlash::Action").action_id("ViewPhoto").build
    let resource =
 EntityIdentifier::builder().entity_id("VacationPhoto94.jpg").entity_type("PhotoFlash::Phot
    let auth = authorize(&client, &policy_store.policy_store_id, &principal,
 &action, &resource).await;
    println!(
        "Decision: {:?}",
        auth.decision
        );
        println!(
        "Policy ID: {:?}",
        auth.determining_policies
        );
     Ok(())
}
```

3. Führen Sie den Code aus, indem Sie `cargo run` ihn in das Terminal eingeben.

Wenn der Code korrekt ausgeführt wird, wird das Terminal angezeigt, `Decision: Allow` gefolgt von der Richtlinien-ID der betreffenden Richtlinie. Das bedeutet, dass Sie erfolgreich einen Richtlinienspeicher erstellt und ihn mit dem AWS SDK für Rust getestet haben.

Bereinigen von -Ressourcen

Nachdem Sie Ihren Richtlinienspeicher durchsucht haben, löschen Sie ihn.

Um einen Richtlinienspeicher zu löschen

Sie können einen Richtlinienspeicher löschen, indem Sie den `delete-policy-store` Vorgang durch die Policy Store-ID *`PSEXAMPLEabcdefg111111`* ersetzen, die Sie löschen möchten.

```
$ aws verifiedpermissions delete-policy-store \
    --policy-store-id PSEXAMPLEabcdefg111111
```

Bei Erfolg erzeugt dieser Befehl keine Ausgabe.

API-verknüpfte Richtlinienspeicher

Ein häufiger Anwendungsfall ist die Verwendung von Amazon Verified Permissions zur Autorisierung des Benutzerzugriffs auf APIs Hosted on Amazon API Gateway. Mithilfe eines Assistenten in der AWS Konsole können Sie rollenbasierte Zugriffsrichtlinien für Benutzer erstellen, die in Amazon Cognito oder einem beliebigen OIDC-Identitätsanbieter (IdP) verwaltet werden, und einen AWS Lambda Authorizer bereitstellen, der Verified Permissions aufruft, um diese Richtlinien auszuwerten.

Um den Assistenten abzuschließen, wählen Sie Setup with API Gateway and an Identity Provider, wenn Sie einen neuen Richtlinienspeicher erstellen, und folgen Sie den Schritten.

Es wird ein API-verknüpfter Richtlinienspeicher erstellt, der Ihr Autorisierungsmodell und Ihre Ressourcen für Autorisierungsanfragen bereitstellt. Der Richtlinienspeicher verfügt über eine Identitätsquelle und einen Lambda-Autorisierer, der API Gateway mit verifizierten Berechtigungen verbindet. Sobald der Richtlinienspeicher erstellt wurde, können Sie API-Anfragen auf der Grundlage der Gruppenmitgliedschaften der Benutzer autorisieren. Verifizierte Berechtigungen können beispielsweise nur Benutzern Zugriff gewähren, die Mitglieder der `Directors` Gruppe sind.

Wenn Ihre Anwendung wächst, können Sie mithilfe der Cedar-Richtliniensprache eine detaillierte Autorisierung mit Benutzerattributen und OAuth 2.0-Bereichen implementieren. Mit Verified Permissions können Sie beispielsweise nur Benutzern Zugriff gewähren, die über ein `email` Attribut in der Domain verfügen. `mycompany.co.uk`

Nachdem Sie das Autorisierungsmodell für Ihre API eingerichtet haben, sind Sie weiterhin dafür verantwortlich, Benutzer zu authentifizieren und API-Anfragen in Ihrer Anwendung zu generieren sowie Ihren Richtlinienspeicher zu verwalten.

Eine Demo finden Sie unter Amazon Verified Permissions — Quick Start Overview und Demo auf dem Amazon Web Services YouTube Kanal.

Themen

- Wie verifizierte Berechtigungen API-Anfragen autorisieren
- Überlegungen zu API-verknüpften Policy-Stores
- Attributbasierte Zugriffskontrolle (ABAC) hinzufügen
- Übergang zur Produktion mit AWS CloudFormation
- Fehlerbehebung bei API-verknüpften Policy-Stores

> **Important**
>
> Richtlinienspeicher, die Sie mit der Option „Mit API Gateway einrichten“ und einer Identitätsquelle in der Konsole „Verifizierte Berechtigungen“ erstellen, sind nicht für die sofortige Bereitstellung in der Produktion vorgesehen. Stellen Sie mit Ihrem ersten Richtlinienspeicher Ihr Autorisierungsmodell fertig und exportieren Sie die Ressourcen des Richtlinienspeichers in CloudFormation. Stellen Sie verifizierte Berechtigungen programmgesteuert mit dem AWS Cloud Development Kit (CDK) für die Produktion bereit. Weitere Informationen finden Sie unter Übergang zur Produktion mit AWS CloudFormation.

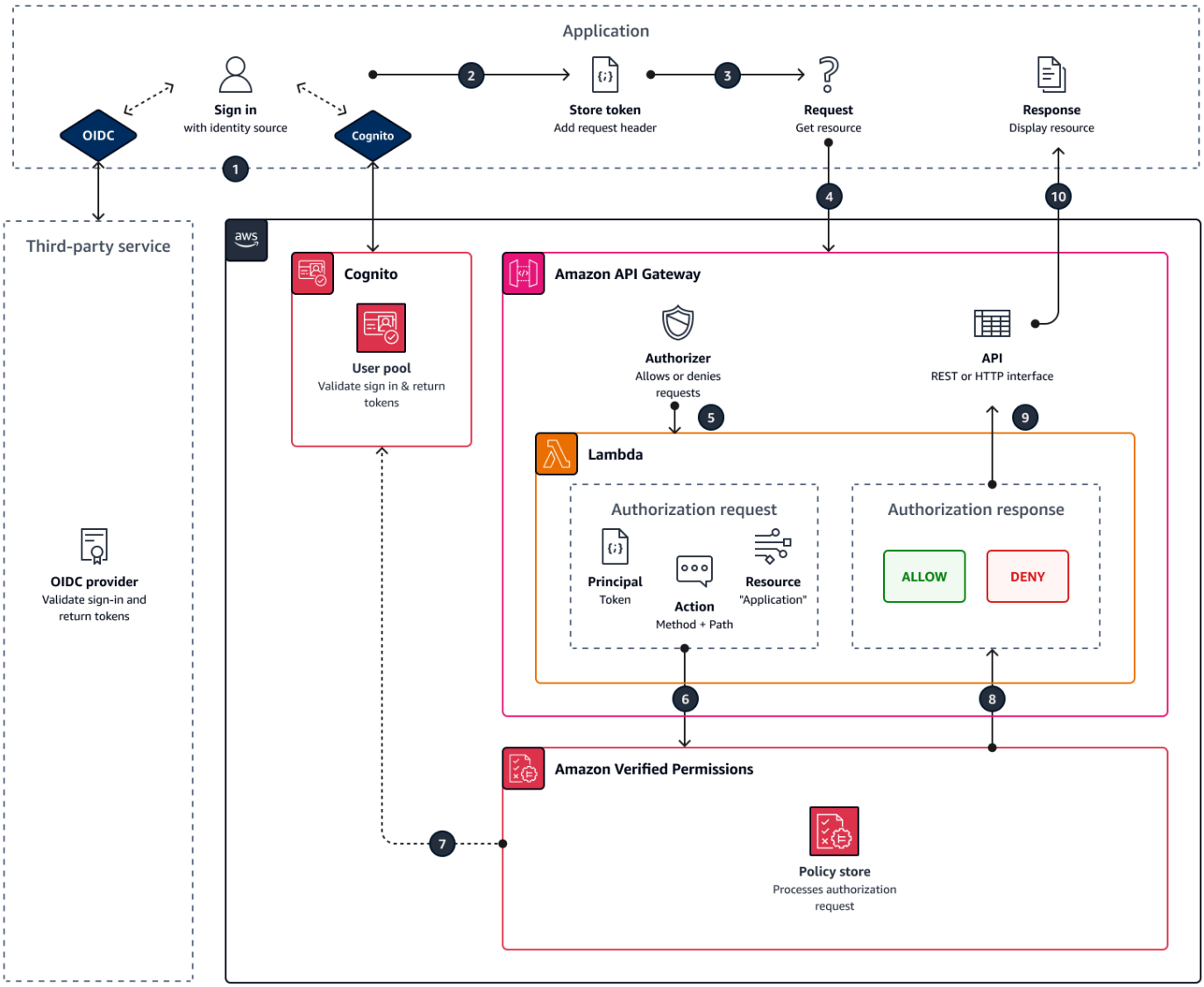
In einem Richtlinienspeicher, der mit einer API und einer Identitätsquelle verknüpft ist, präsentiert Ihre Anwendung ein Benutzerpool-Token in einem Autorisierungsheader, wenn sie eine Anfrage an die API stellt. Die Identitätsquelle Ihres Richtlinienspeichers ermöglicht die Tokenvalidierung für verifizierte Berechtigungen. Das Token bildet die `principal` Eingangs-Autorisierungsanfragen mit der IsAuthorizedWithTokenAPI. Verified Permissions erstellt Richtlinien rund um die Gruppenzugehörigkeit Ihrer Benutzer, wie sie in einem Gruppenanspruch in Form von Identitäts- (ID) und Zugriffstoken dargestellt werden, z. B. `cognito:groups` für Benutzerpools. Ihre API verarbeitet das Token aus Ihrer Anwendung in einem Lambda-Autorisierer und leitet es zur Autorisierungsentscheidung an Verified Permissions weiter. Wenn Ihre API die Autorisierungsentscheidung vom Lambda-Autorisierer erhält, leitet sie die Anfrage an Ihre Datenquelle weiter oder lehnt die Anfrage ab.

Komponenten der Identitätsquelle und der API-Gateway-Autorisierung mit verifizierten Berechtigungen

- Ein Amazon Cognito Cognito-Benutzerpool oder OIDC-IdP, der Benutzer authentifiziert und gruppiert. Benutzertoken füllen die Gruppenmitgliedschaft und den Prinzipal oder Kontext, den Verified Permissions auswertet, in Ihrem Richtlinienspeicher aus.
- Eine API-Gateway-REST-API. Verified Permissions definiert beispielsweise Aktionen anhand von API-Pfaden und API-Methoden`MyAPI::Action::get /photo`.
- Eine Lambda-Funktion und ein Lambda-Authorizer für Ihre API. Die Lambda-Funktion nimmt Bearer-Token aus Ihrem Benutzerpool entgegen, fordert die Autorisierung von Verified Permissions an und gibt eine Entscheidung an API Gateway zurück. Der Workflow Setup with API Gateway and an Identity Source erstellt diesen Lambda-Authorizer automatisch für Sie.
- Ein Richtlinienspeicher für verifizierte Berechtigungen. Die Identitätsquelle für den Richtlinienspeicher ist Ihr Amazon Cognito Cognito-Benutzerpool oder Ihre OIDC-Anbietergruppe. Das Policy Store-Schema spiegelt die Konfiguration Ihrer API wider, und die Richtlinien verknüpfen Benutzergruppen mit zulässigen API-Aktionen.
- Eine Anwendung, die Benutzer bei Ihrem IdP authentifiziert und Tokens an API-Anfragen anhängt.

Wie verifizierte Berechtigungen API-Anfragen autorisieren

Wenn Sie einen neuen Richtlinienspeicher erstellen und die Option Mit API Gateway und einer Identitätsquelle einrichten auswählen, erstellt Verified Permissions ein Richtlinienspeicherschema und Richtlinien. Das Schema und die Richtlinien spiegeln die API-Aktionen und die Benutzergruppen wider, die Sie zur Durchführung der Aktionen autorisieren möchten. Verified Permissions erstellt auch die Lambda-Funktion und den Autorisierer.



1. Ihr Benutzer meldet sich mit Ihrer Anwendung über Amazon Cognito oder einen anderen OIDC-IdP an. Der IdP gibt ID- und Zugriffstoken mit den Benutzerinformationen aus.
2. Ihre Anwendung speichert die JWTs. Weitere Informationen finden Sie unter Verwenden von Token mit Benutzerpools im Amazon Cognito Developer Guide.
3. Ihr Benutzer fordert Daten an, die Ihre Anwendung von einer externen API abrufen muss.
4. Ihre Anwendung fordert Daten von einer REST-API in API Gateway an. Sie hängt eine ID oder ein Zugriffstoken als Anforderungsheader an.
5. Wenn Ihre API über einen Cache für die Autorisierungsentscheidung verfügt, gibt sie die vorherige Antwort zurück. Wenn das Caching deaktiviert ist oder die API keinen aktuellen Cache hat, übergibt API Gateway die Anforderungsparameter an einen tokenbasierten Lambda-Authorizer.

6. Die Lambda-Funktion sendet mit der IsAuthorizedWithTokenAPI eine Autorisierungsanfrage an einen Richtlinienspeicher für verifizierte Berechtigungen. Die Lambda-Funktion übergibt die Elemente einer Autorisierungsentscheidung:
   a. Das Token des Benutzers als Principal.
   b. Die API-Methode kombiniert mit dem API-Pfad`GetPhoto`, z. B. als Aktion.
   c. Der Begriff `Application` als Ressource.
7. Verified Permissions validiert das Token. Weitere Informationen zur Validierung von Amazon Cognito-Token finden Sie unter Authorization with Amazon Verified Permissions im Amazon Cognito Developer Guide.
8. Verified Permissions bewertet die Autorisierungsanfrage anhand der Richtlinien in Ihrem Richtlinienspeicher und gibt eine Autorisierungsentscheidung zurück.
9. Der Lambda-Autorisierer gibt eine `Allow` `Deny` Oder-Antwort an API Gateway zurück.
10. Die API gibt Daten oder eine `ACCESS_DENIED` Antwort an Ihre Anwendung zurück. Ihre Anwendung verarbeitet die Ergebnisse der API-Anfrage und zeigt sie an.

Überlegungen zu API-verknüpften Policy-Stores

Wenn Sie in der Konsole „Verified Permissions“ einen API-verknüpften Richtlinienspeicher erstellen, erstellen Sie einen Test für eine spätere Produktionsbereitstellung. Bevor Sie zur Produktion übergehen, richten Sie eine feste Konfiguration für Ihre API und Ihren Benutzerpool ein. Berücksichtigen Sie die folgenden Faktoren:

API Gateway speichert Antworten im Cache

In API-verknüpften Richtlinienspeichern erstellt Verified Permissions einen Lambda-Autorisierer mit einer Autorisierungs-Caching-TTL von 120 Sekunden. Sie können diesen Wert anpassen oder das Caching in Ihrem Authorizer deaktivieren. In einem Autorisierer mit aktiviertem Caching gibt Ihr Autorisierer jedes Mal dieselbe Antwort zurück, bis die TTL abläuft. Dadurch kann die tatsächliche Lebensdauer von Benutzerpool-Token um eine Dauer verlängert werden, die der Caching-TTL der angeforderten Phase entspricht.

Amazon Cognito Cognito-Gruppen können wiederverwendet werden

Amazon Verified Permissions bestimmt die Gruppenmitgliedschaft von Benutzern aus dem Benutzerpool anhand des `cognito:groups` Antrags in der ID oder dem Zugriffstoken eines Benutzers. Der Wert dieses Anspruchs besteht aus einer Reihe von benutzerfreundlichen Namen

der Benutzerpoolgruppen, denen der Benutzer angehört. Sie können Benutzerpoolgruppen keinen eindeutigen Bezeichner zuordnen.

Benutzerpoolgruppen, die Sie löschen und mit demselben Namen neu erstellen, werden in Ihrem Richtlinienspeicher als dieselbe Gruppe angezeigt. Wenn Sie eine Gruppe aus einem Benutzerpool löschen, löschen Sie alle Verweise auf die Gruppe aus Ihrem Richtlinienspeicher.

Von der API abgeleiteter Namespace und Schema sind point-in-time

Verified Permissions erfasst Ihre API zu einem bestimmten Zeitpunkt: Ihre API wird nur abgefragt, wenn Sie Ihren Richtlinienspeicher erstellen. Wenn sich das Schema oder der Name Ihrer API ändert, müssen Sie Ihren Richtlinienspeicher und Ihren Lambda-Autorisierer aktualisieren oder einen neuen API-verknüpften Richtlinienspeicher erstellen. Verified Permissions leitet den Namespace für den Richtlinienspeicher vom Namen Ihrer API ab.

Lambda Lambda-Funktion hat keine VPC-Konfiguration

Die Lambda-Funktion, die Verified Permissions für Ihren API-Authorizer erstellt, wird in der Standard-VPC gestartet. Standardmäßig. APIs deren Netzwerkzugriff auf privat beschränkt ist, VPCs können nicht mit der Lambda-Funktion kommunizieren, die Zugriffsanfragen mit verifizierten Berechtigungen autorisiert.

Verified Permissions stellt autorisierte Ressourcen bereit in CloudFormation

Um einen API-verknüpften Richtlinienspeicher zu erstellen, müssen Sie sich bei der Verified Permissions-Konsole mit einem AWS Prinzipal mit hohen Rechten anmelden. Dieser Benutzer stellt einen AWS CloudFormation Stapel bereit, der Ressourcen aus mehreren zusammensetzt. AWS-Services Dieser Principal muss über die Berechtigung verfügen, Ressourcen in Verified Permissions IAM, Lambda und API Gateway hinzuzufügen und zu ändern. Es hat sich bewährt, diese Anmeldeinformationen nicht mit anderen Administratoren in Ihrer Organisation zu teilen.

Einen Überblick über die Ressourcen, die Verified Permissions erstellt, finden Übergang zur Produktion mit AWS CloudFormation Sie unter.

Attributbasierte Zugriffskontrolle (ABAC) hinzufügen

Eine typische Authentifizierungssitzung mit einem IdP gibt ID- und Zugriffstoken zurück. Sie können jeden dieser Tokentypen als Bearer-Token in Anwendungsanfragen an Ihre API übergeben. Je nachdem, welche Optionen Sie bei der Erstellung Ihres Richtlinienspeichers ausgewählt haben, erwartet Verified Permissions einen der beiden Token-Typen. Beide Typen enthalten Informationen

über die Gruppenmitgliedschaft des Benutzers. Weitere Informationen zu Tokentypen in Amazon Cognito finden Sie unter Verwenden von Token mit Benutzerpools im Amazon Cognito Developer Guide.

Nachdem Sie einen Richtlinienspeicher erstellt haben, können Sie Richtlinien hinzufügen und erweitern. Beispielsweise können Sie Ihren Richtlinien neue Gruppen hinzufügen, wenn Sie sie Ihrem Benutzerpool hinzufügen. Da Ihr Richtlinienspeicher bereits weiß, wie Ihr Benutzerpool Gruppen in Tokens präsentiert, können Sie eine Reihe von Aktionen für jede neue Gruppe mit einer neuen Richtlinie zulassen.

Möglicherweise möchten Sie auch das gruppenbasierte Modell der Richtlinienbewertung um ein genaueres Modell erweitern, das auf Benutzereigenschaften basiert. Benutzerpool-Token enthalten zusätzliche Benutzerinformationen, die zu Autorisierungsentscheidungen beitragen können.

ID-Token

ID-Token stellen die Attribute eines Benutzers dar und bieten ein hohes Maß an detaillierter Zugriffskontrolle. Um E-Mail-Adressen, Telefonnummern oder benutzerdefinierte Attribute wie Abteilung und Manager auszuwerten, werten Sie das ID-Token aus.

Zugriffstoken

Zugriffstoken stellen die Berechtigungen eines Benutzers mit einem Geltungsbereich von OAuth 2.0 dar. Um eine Autorisierungsebene hinzuzufügen oder Anfragen für zusätzliche Ressourcen einzurichten, bewerten Sie das Zugriffstoken. Sie können beispielsweise überprüfen, ob ein Benutzer zu den entsprechenden Gruppen gehört und über einen solchen Geltungsbereich verfügt`PetStore.read`, der in der Regel den Zugriff auf die API autorisiert. Benutzerpools können Tokens mit Ressourcenservern und mit Token-Anpassungen zur Laufzeit benutzerdefinierte Bereiche hinzufügen.

Sehen Sie sich Zuordnen von Identitätsanbieter-Tokens zum Schema zum Beispiel Richtlinien an, die Ansprüche in ID- und Zugriffstoken verarbeiten.

Übergang zur Produktion mit AWS CloudFormation

API-verknüpfte Richtlinienspeicher sind eine Möglichkeit, schnell ein Autorisierungsmodell für eine API Gateway zu erstellen. Sie dienen als Testumgebung für die Autorisierungskomponente Ihrer Anwendung. Nachdem Sie Ihren Testrichtlinienspeicher erstellt haben, sollten Sie Zeit damit verbringen, die Richtlinien, das Schema und den Lambda-Autorisierer zu verfeinern.

Möglicherweise passen Sie die Architektur Ihrer API an, sodass entsprechende Anpassungen an Ihrem Richtlinienspeicher-Schema und Ihren Richtlinien erforderlich sind. API-verknüpfte Richtlinienspeicher aktualisieren ihr Schema nicht automatisch über die API-Architektur — Verified Permissions fragt die API nur ab, wenn Sie einen Richtlinienspeicher erstellen. Wenn sich Ihre API ausreichend ändert, müssen Sie den Vorgang möglicherweise mit einem neuen Richtlinienspeicher wiederholen.

Wenn Ihre Anwendung und Ihr Autorisierungsmodell bereit für die Bereitstellung in der Produktion sind, integrieren Sie den API-verknüpften Richtlinienspeicher, den Sie entwickelt haben, in Ihre Automatisierungsprozesse. Als bewährte Methode empfehlen wir, das Richtlinienspeicherschema und die Richtlinien in eine AWS CloudFormation Vorlage zu exportieren, die Sie in anderen AWS-Konten Bereichen bereitstellen können. AWS-Regionen

Die Ergebnisse des API-verknüpften Richtlinienspeicher-Prozesses sind ein erster Richtlinienspeicher und ein Lambda-Autorisierer. Der Lambda-Autorisierer hat mehrere abhängige Ressourcen. Verified Permissions stellt diese Ressourcen in einem automatisch generierten Stack bereit. CloudFormation Für die Bereitstellung in der Produktion müssen Sie den Richtlinienspeicher und die Lambda-Autorisierungsressourcen in einer Vorlage zusammenfassen. Ein API-verknüpfter Richtlinienspeicher besteht aus den folgenden Ressourcen:

1. AWS::VerifiedPermissions::PolicyStore: Kopieren Sie Ihr Schema in das `SchemaDefinition` Objekt. "Escape-Zeichen als\".
2. AWS::VerifiedPermissions::IdentitySource: Kopieren Sie Werte aus der Ausgabe von GetIdentitySource aus Ihrem Testrichtlinienspeicher und ändern Sie sie nach Bedarf.
3. Eine oder mehrere der folgenden AWS::VerifiedPermissions::PolicyOptionen: Kopieren Sie Ihre Richtlinienerklärung in das `Definition` Objekt. "Escape-Zeichen als\".
4. AWS: :Lambda: :Function,: AWS:: :Rolle,IAM::: :Policy,AWS::IAM: :Authorizer,AWS ApiGateway AWS::Lambda::Permission

Die folgende Vorlage ist ein Beispiel für einen Richtlinienspeicher. Sie können die Lambda-Authorizer-Ressourcen aus Ihrem vorhandenen Stack an diese Vorlage anhängen.

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "MyExamplePolicyStore": {
            "Type": "AWS::VerifiedPermissions::PolicyStore",
            "Properties": {
```

```
                "ValidationSettings": {
                    "Mode": "STRICT"
                },
                "Description": "ApiGateway: PetStore/test",
                "Schema": {
                    "CedarJson": "{\"PetStore\":{\"actions\":{\"get /pets\":
{\"appliesTo\":{\"principalTypes\":[\"User\"],\"resourceTypes\":[\"Application\"],
\"context\":{\"type\":\"Record\",\"attributes\":{}}}},\"get /\":{\"appliesTo\":
{\"principalTypes\":[\"User\"],\"resourceTypes\":[\"Application\"],\"context\":{\"type
\":\"Record\",\"attributes\":{}}}},\"get /pets/{petId}\":{\"appliesTo\":{\"context
\":{\"type\":\"Record\",\"attributes\":{}},\"resourceTypes\":[\"Application\"],
\"principalTypes\":[\"User\"]}},\"post /pets\":{\"appliesTo\":{\"principalTypes\":
[\"User\"],\"resourceTypes\":[\"Application\"],\"context\":{\"type\":\"Record\",
\"attributes\":{}}}}},\"entityTypes\":{\"Application\":{\"shape\":{\"type\":\"Record\",
\"attributes\":{}}},\"User\":{\"memberOfTypes\":[\"UserGroup\"],\"shape\":{\"attributes
\":{},\"type\":\"Record\"}},\"UserGroup\":{\"shape\":{\"type\":\"Record\",\"attributes
\":{}}}}}}"
                }
            }
        },
        "MyExamplePolicy": {
            "Type": "AWS::VerifiedPermissions::Policy",
            "Properties": {
                "Definition": {
                    "Static": {
                        "Description": "Policy defining permissions for testgroup
 cognito group",
                        "Statement": "permit(\nprincipal in PetStore::UserGroup::
\"us-east-1_EXAMPLE|testgroup\",\naction in [\n  PetStore::Action::\"get /\",
\n  PetStore::Action::\"post /pets\",\n  PetStore::Action::\"get /pets\",\n
 PetStore::Action::\"get /pets/{petId}\"\n],\nresource);"
                    }
                },
                "PolicyStoreId": {
                    "Ref": "MyExamplePolicyStore"
                }
            },
            "DependsOn": [
                "MyExamplePolicyStore"
            ]
        },
        "MyExampleIdentitySource": {
            "Type": "AWS::VerifiedPermissions::IdentitySource",
            "Properties": {
```

```
                "Configuration": {
                    "CognitoUserPoolConfiguration": {
                        "ClientIds": [
                            "1example23456789"
                        ],
                        "GroupConfiguration": {
                            "GroupEntityType": "PetStore::UserGroup"
                        },
                        "UserPoolArn": "arn:aws:cognito-idp:us-
east-1:123456789012:userpool/us-east-1_EXAMPLE"
                    }
                },
                "PolicyStoreId": {
                    "Ref": "MyExamplePolicyStore"
                },
                "PrincipalEntityType": "PetStore::User"
            },
            "DependsOn": [
                "MyExamplePolicyStore"
            ]
        }
    }
}
```

Fehlerbehebung bei API-verknüpften Policy-Stores

Verwenden Sie die Informationen hier, um häufig auftretende Probleme beim Erstellen von mit der Amazon Verified Permissions API verknüpften Policy Stores zu diagnostizieren und zu beheben.

Themen

- Ich habe meine Richtlinie aktualisiert, aber die Autorisierungsentscheidung hat sich nicht geändert
- Ich habe den Lambda-Autorisierer an meine API angehängt, aber er generiert keine Autorisierungsanfragen
- Ich habe eine unerwartete Autorisierungsentscheidung erhalten und möchte die Autorisierungslogik überprüfen
- Ich möchte Logs von meinem Lambda-Authorizer finden
- Mein Lambda-Autorisierer existiert nicht
- Meine API befindet sich in einer privaten VPC und kann den Authorizer nicht aufrufen
- Ich möchte zusätzliche Benutzerattribute in meinem Autorisierungsmodell verarbeiten

- Ich möchte neue Aktionen, Aktionskontext-Attribute oder Ressourcenattribute hinzufügen

Ich habe meine Richtlinie aktualisiert, aber die Autorisierungsentscheidung hat sich nicht geändert

Standardmäßig konfiguriert Verified Permissions den Lambda-Autorisierer so, dass Autorisierungsentscheidungen 120 Sekunden lang zwischengespeichert werden. Versuchen Sie es nach zwei Minuten erneut, oder deaktivieren Sie den Cache auf Ihrem Authorizer. Weitere Informationen finden Sie unter Aktivieren von API-Caching zur Verbesserung der Reaktionsfähigkeit im Amazon API Gateway Developer Guide.

Ich habe den Lambda-Autorisierer an meine API angehängt, aber er generiert keine Autorisierungsanfragen

Um mit der Bearbeitung von Anfragen zu beginnen, müssen Sie die API-Stufe bereitstellen, an die Sie Ihren Autorisierer angehängt haben. Weitere Informationen finden Sie unter Bereitstellen einer REST-API im Amazon API Gateway Developer Guide.

Ich habe eine unerwartete Autorisierungsentscheidung erhalten und möchte die Autorisierungslogik überprüfen

Der API-verknüpfte Policy Store-Prozess erstellt eine Lambda-Funktion für Ihren Autorisierer. Verified Permissions integriert die Logik Ihrer Autorisierungsentscheidungen automatisch in die Autorisierungsfunktion. Nachdem Sie Ihren Richtlinienspeicher erstellt haben, können Sie zurückgehen, um die Logik in der Funktion zu überprüfen und zu aktualisieren.

Um Ihre Lambda-Funktion von der AWS CloudFormation Konsole aus zu finden, klicken Sie auf der Übersichtsseite Ihres neuen Policy-Stores auf die Schaltfläche Bereitstellung überprüfen.

Sie können Ihre Funktion auch in der AWS Lambda Konsole finden. Navigieren Sie zur Konsole in Ihrem Richtlinienspeicher und suchen Sie nach einem Funktionsnamen mit dem Präfix`AVPAuthorizerLambda`. AWS-Region Wenn Sie mehr als einen API-verknüpften Richtlinienspeicher erstellt haben, verwenden Sie die Uhrzeit der letzten Änderung Ihrer Funktionen, um sie mit der Erstellung des Richtlinienspeichers zu korrelieren.

Ich möchte Logs von meinem Lambda-Authorizer finden

Lambda-Funktionen sammeln Metriken und protokollieren ihre Aufrufergebnisse in Amazon. CloudWatch Um Ihre Logs zu überprüfen, suchen Sie Ihre Funktion in der Lambda-Konsole und

wählen Sie die Registerkarte Überwachen. Wählen Sie CloudWatch Protokolle anzeigen aus und überprüfen Sie die Einträge in der Protokollgruppe.

Weitere Informationen zu Lambda-Funktionsprotokollen finden Sie unter Using Amazon CloudWatch Logs with AWS Lambda im AWS Lambda Developer Guide.

Mein Lambda-Autorisierer existiert nicht

Nachdem Sie die Einrichtung eines API-verknüpften Richtlinienspeichers abgeschlossen haben, müssen Sie den Lambda-Authorizer an Ihre API anhängen. Wenn Sie Ihren Autorisierer in der API Gateway Gateway-Konsole nicht finden können, sind die zusätzlichen Ressourcen für Ihren Richtlinienspeicher möglicherweise ausgefallen oder noch nicht bereitgestellt. API-verknüpfte Policy-Stores stellen diese Ressourcen in einem Stapel bereit. AWS CloudFormation

Unter Verifizierte Berechtigungen wird am Ende des Erstellungsvorgangs ein Link mit der Bezeichnung Bereitstellung prüfen angezeigt. Wenn Sie diesen Bildschirm bereits verlassen haben, rufen Sie die CloudFormation Konsole auf und suchen Sie in den letzten Stacks nach einem Namen, dem das Präfix vorangestellt ist. `AVPAuthorizer-<policy store ID>` CloudFormation bietet wertvolle Informationen zur Fehlerbehebung in der Ausgabe einer Stack-Bereitstellung.

Hilfe zur Fehlerbehebung bei CloudFormation Stacks finden Sie unter Problembehandlung CloudFormation im AWS CloudFormation Benutzerhandbuch.

Meine API befindet sich in einer privaten VPC und kann den Authorizer nicht aufrufen

Verified Permissions unterstützt keinen Zugriff auf Lambda-Autorisierer über VPC-Endpunkte. Sie müssen einen Netzwerkpfad zwischen Ihrer API und der Lambda-Funktion öffnen, die als Autorisierer dient.

Ich möchte zusätzliche Benutzerattribute in meinem Autorisierungsmodell verarbeiten

Der API-verknüpfte Policy-Store-Prozess leitet Richtlinien für verifizierte Berechtigungen aus dem Anspruch der Gruppe in Benutzer-Tokens ab. Um Ihr Autorisierungsmodell zu aktualisieren und zusätzliche Benutzerattribute zu berücksichtigen, integrieren Sie diese Attribute in Ihre Richtlinien.

Sie können viele Ansprüche in ID- und Zugriffstoken aus Amazon Cognito Cognito-Benutzerpools den Richtlinienerklärungen für verifizierte Berechtigungen zuordnen. Beispielsweise haben die meisten Benutzer einen `email` Anspruch in ihrem ID-Token. Weitere Informationen zum Hinzufügen von Ansprüchen aus Ihrer Identitätsquelle zu Richtlinien finden Sie unterZuordnen von Identitätsanbieter-Tokens zum Schema.

Ich möchte neue Aktionen, Aktionskontext-Attribute oder Ressourcenattribute hinzufügen

Ein API-verknüpfter Richtlinienspeicher und der Lambda-Autorisierer, den er erstellt, sind eine Ressource. point-in-time Sie geben den Status Ihrer API zum Zeitpunkt der Erstellung wieder. Das Policy-Store-Schema weist Aktionen weder Kontext-Attribute noch Attribute oder übergeordnete Elemente der `Application` Standardressource zu.

Wenn Sie Ihrer API Aktionen — Pfade und Methoden — hinzufügen, müssen Sie Ihren Richtlinienspeicher aktualisieren, damit er über die neuen Aktionen informiert ist. Sie müssen auch Ihren Lambda-Autorisierer aktualisieren, um Autorisierungsanfragen für die neuen Aktionen zu verarbeiten. Sie können mit einem neuen Richtlinienspeicher erneut beginnen oder Ihren vorhandenen Richtlinienspeicher aktualisieren.

Suchen Sie nach Ihrer Funktion, um Ihren vorhandenen Richtlinienspeicher zu aktualisieren. Untersuchen Sie die Logik in der automatisch generierten Funktion und aktualisieren Sie sie, um die neuen Aktionen, Attribute oder den Kontext zu verarbeiten. Bearbeiten Sie dann Ihr Schema so, dass es die neuen Aktionen und Attribute enthält.

Policy-Stores löschen

Sie können Amazon Verified Permissions Policy Stores mit dem AWS Management Console oder dem löschen AWS CLI. Durch das Löschen eines Richtlinienspeichers werden das Schema und alle Richtlinien im Richtlinienspeicher dauerhaft gelöscht.

Der Löschschutz verhindert das versehentliche Löschen eines Richtlinienspeichers. Der Löschschutz ist für alle neuen Richtlinienspeicher aktiviert, die über den erstellt wurden AWS Management Console. Im Gegensatz dazu ist er für alle Richtlinienspeicher deaktiviert, die über einen API- oder SDK-Aufruf erstellt wurden.

Möglicherweise möchten Sie Richtlinienspeicher aus den folgenden Gründen löschen:

- Sie haben das Kontingent verfügbarer Policy Stores in einer bestimmten Region erreicht. Weitere Informationen finden Sie unter Kontingente für Ressourcen.
- Sie unterstützen keinen Mandanten mehr in einer Mehrmandantenanwendung und benötigen daher diesen Richtlinienspeicher nicht mehr.

AWS Management Console

Um einen Richtlinienspeicher zu löschen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Settings aus.
3. Wählen Sie Diesen Richtlinienspeicher löschen aus.
4. Geben Sie `delete` das Textfeld ein und wählen Sie Löschen.

> **Note**
>
> Wenn der Löschschutz aktiviert ist, müssen Sie ihn deaktivieren, bevor Sie Löschen wählen können. Um ihn zu deaktivieren, wählen Sie Löschschutz deaktivieren.

AWS CLI

Um einen Richtlinienspeicher zu löschen

Sie können einen Richtlinienspeicher löschen, indem Sie den `delete-policy-store` Vorgang durch die Policy Store-ID *`PSEXAMPLEabcdefg111111`* ersetzen, die Sie löschen möchten.

```
$ aws verifiedpermissions delete-policy-store \
    --policy-store-id PSEXAMPLEabcdefg111111
```

Bei Erfolg erzeugt dieser Befehl keine Ausgabe.

> **Note**
>
> Wenn der Löschschutz für diesen Richtlinienspeicher aktiviert ist, müssen Sie zuerst den `update-policy-store` Vorgang ausführen und den Löschschutz deaktivieren.
>
> ```
> aws verifiedpermissions update-policy-store \
>     --deletion-protection "DISABLED" \
>     --policy-store-id PSEXAMPLEabcdefg111111
> ```

Speicherschema für Richtlinien von Amazon Verified Permissions

Ein Schema ist eine Deklaration der Struktur der Entitätstypen, die von Ihrer Anwendung unterstützt werden, und der Aktionen, die Ihre Anwendung in Autorisierungsanfragen bereitstellen kann. Informationen zum Unterschied zwischen der Art und Weise, wie Verified Permissions und Cedar Schemas handhaben, finden Sie unterSchema-Unterstützung.

Weitere Informationen finden Sie unter Cedar Schema Format im Cedar Policy Language Reference Guide.

> **Note**
>
> Die Verwendung von Schemas in Verified Permissions ist optional, wird aber für Produktionssoftware dringend empfohlen. Wenn Sie eine neue Richtlinie erstellen, kann Verified Permissions das Schema verwenden, um die Entitäten und Attribute zu überprüfen, auf die im Bereich und in den Bedingungen verwiesen wird, um Tippfehler und Fehler in Richtlinien zu vermeiden, die zu verwirrendem Systemverhalten führen können. Wenn Sie die Richtlinienvalidierung aktivieren, müssen alle neuen Richtlinien dem Schema entsprechen.

AWS Management Console

Ein Schema erstellen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Schema aus.
3. Wählen Sie Create schema (Schema erstellen) aus.

AWS CLI

Um ein neues Schema einzureichen oder ein vorhandenes Schema zu überschreiben, verwenden Sie den AWS CLI.

Sie können einen Richtlinienspeicher erstellen, indem Sie einen AWS CLI Befehl ausführen, der dem folgenden Beispiel ähnelt.

Stellen Sie sich ein Schema vor, das den folgenden Cedar-Inhalt enthält:

```
{
    "MySampleNamespace": {
        "actions": {
            "remoteAccess": {
                "appliesTo": {
                    "principalTypes": [ "Employee" ]
                }
            }
        },
        "entityTypes": {
            "Employee": {
                "shape": {
                    "type": "Record",
                    "attributes": {
                        "jobLevel": {"type": "Long"},
                        "name": {"type": "String"}
                    }
                }
            }
        }
    }
}
```

Sie müssen die JSON-Datei zunächst in eine einzeilige Zeichenfolge umwandeln und ihr eine Deklaration ihres Datentyps voranstellen:`cedarJson`. Im folgenden Beispiel wird der folgende Inhalt einer `schema.json` Datei verwendet, die die Escape-Version des JSON-Schemas enthält.

> **Note**
>
> Das Beispiel hier ist aus Gründen der Lesbarkeit mit einem Zeilenumbruch versehen. Sie müssen die gesamte Datei in einer einzigen Zeile haben, damit der Befehl sie akzeptiert.

```
{"cedarJson": "{\"MySampleNamespace\": {\"actions\": {\"remoteAccess\": {\"appliesTo
\":
{\"principalTypes\": [\"Employee\"]}}},\"entityTypes\": {\"Employee\": {\"shape\":
{\"attributes\": {\"jobLevel\": {\"type\": \"Long\"},\"name\": {\"type\": \"String
\"}},
\"type\": \"Record\"}}}}}"}
```

```
$ aws verifiedpermissions put-schema \
    --definition file://schema.json \
    --policy-store PSEXAMPLEabcdefg111111
{
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "namespaces": [
        "MySampleNamespace"
    ],
    "createdDate": "2023-07-17T21:07:43.659196+00:00",
    "lastUpdatedDate": "2023-08-16T17:03:53.081839+00:00"
}
```

AWS SDKs

Sie können mithilfe der `PutSchema` API einen Richtlinienspeicher erstellen. Weitere Informationen finden Sie PutSchemaim Referenzhandbuch zur Amazon Verified Permissions API.

Policy-Store-Schemas bearbeiten

Wenn Sie in der Amazon Verified Permissions-Konsole Schema auswählen, werden die Entitätstypen und Aktionen angezeigt, aus denen Ihr Schema besteht. Sie können Ihr Schema entweder im visuellen Modus oder im JSON-Modus anzeigen und bearbeiten. Im visuellen Modus können Sie das Schema aktualisieren, indem Sie mithilfe verschiedener Assistenten neue Typen und Aktionen hinzufügen. Im JSON-Modus können Sie direkt im JSON-Editor mit der Aktualisierung des JSON-Codes des Schemas beginnen.

Visual Mode

Der visuelle Schema-Editor beginnt mit einer Reihe von Diagrammen, die die Beziehungen zwischen den Entitäten in Ihrem Schema veranschaulichen. Wählen Sie Erweitern, um Ihre Ansicht der Diagramme zu maximieren. Es sind zwei Diagramme verfügbar:

- Aktionsdiagramm — In der Diagrammansicht Aktionen werden die Typen von Prinzipalen aufgeführt, die Sie in Ihrem Richtlinienspeicher konfiguriert haben, die Aktionen, zu deren Ausführung sie berechtigt sind, und die Ressourcen, für die sie Aktionen ausführen können. Die Linien zwischen den Entitäten zeigen, dass Sie in der Lage sind, eine Richtlinie zu erstellen, die es einem Prinzipal ermöglicht, eine Aktion für eine Ressource durchzuführen. Wenn Ihr Aktionsdiagramm keine Beziehung zwischen zwei Entitäten anzeigt, müssen Sie diese Beziehung zwischen ihnen herstellen, bevor Sie sie in Richtlinien zulassen oder verweigern

können. Wählen Sie eine Entität aus, um eine Übersicht über die Eigenschaften zu erhalten, und führen Sie einen Drilldown durch, um alle Details anzuzeigen. Wählen Sie Nach [Aktion | Ressourcentyp | Prinzipaltyp] filtern, um eine Entität in einer Ansicht mit nur ihren eigenen Verbindungen anzuzeigen.

- Entitätstypen-Diagramm — Das Entitätstypen-Diagramm konzentriert sich auf die Beziehungen zwischen Prinzipalen und Ressourcen. Wenn Sie die komplexen verschachtelten Elternbeziehungen in Ihrem Schema verstehen möchten, sehen Sie sich dieses Diagramm an. Zeigen Sie mit der Maus auf eine Entität, um sich die übergeordneten Beziehungen anzusehen, die sie besitzt.

  Unter den Diagrammen befinden sich Listenansichten der Entitätstypen und Aktionen in Ihrem Schema. Die Listenansicht ist nützlich, wenn Sie sofort die Details einer bestimmten Aktion oder eines bestimmten Entitätstyps anzeigen möchten. Wählen Sie eine beliebige Entität aus, um Details anzuzeigen.

Um ein Schema für verifizierte Berechtigungen im visuellen Modus zu bearbeiten

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Schema aus.
3. Wählen Sie den Visuellen Modus. Überprüfen Sie die Entity-Relationship-Diagramme und planen Sie die Änderungen, die Sie an Ihrem Schema vornehmen möchten. Sie können optional nach einer Entität filtern, um deren individuelle Verbindungen zu anderen Entitäten zu untersuchen.
4. Wählen Sie Edit schema (Schema bearbeiten).
5. Geben Sie im Abschnitt Details einen Namespace für Ihr Schema ein.
6. Wählen Sie im Abschnitt Entitätstypen die Option Neuen Entitätstyp hinzufügen aus.
7. Geben Sie den Namen der Entität ein.
8. (Optional) Wählen Sie Übergeordnete Entitäten hinzufügen aus, um übergeordnete Entitäten hinzuzufügen, denen die neue Entität angehört. Um ein übergeordnetes Element zu entfernen, das der Entität hinzugefügt wurde, wählen Sie neben dem Namen der übergeordneten Entität die Option Entfernen aus.
9. Wählen Sie Attribut hinzufügen, um der Entität Attribute hinzuzufügen. Geben Sie den Attributnamen ein und wählen Sie den Attributtyp für jedes Attribut der Entität aus. Verified Permissions verwendet die angegebenen Attributwerte, wenn Richtlinien anhand des Schemas überprüft werden. Wählen Sie aus, ob jedes Attribut erforderlich ist. Um ein Attribut

zu entfernen, das der Entität hinzugefügt wurde, wählen Sie neben dem Attribut die Option Entfernen aus.

10. Wählen Sie Entitätstyp hinzufügen, um die Entität zum Schema hinzuzufügen.
11. Wählen Sie im Abschnitt Aktionen die Option Neue Aktion hinzufügen aus.
12. Geben Sie den Namen der Aktion ein.
13. (Optional) Wählen Sie Ressource hinzufügen aus, um Ressourcentypen hinzuzufügen, für die die Aktion gilt. Um einen Ressourcentyp zu entfernen, der der Aktion hinzugefügt wurde, wählen Sie neben dem Namen des Ressourcentyps die Option Entfernen aus.
14. (Optional) Wählen Sie Prinzipal hinzufügen aus, um einen Prinzipaltyp hinzuzufügen, für den die Aktion gilt. Um einen Prinzipaltyp zu entfernen, der der Aktion hinzugefügt wurde, wählen Sie neben dem Namen des Prinzipaltyps die Option Entfernen aus.
15. Wählen Sie Attribut hinzufügen aus, um Attribute hinzuzufügen, die dem Kontext einer Aktion in Ihren Autorisierungsanfragen hinzugefügt werden können. Geben Sie den Attributnamen ein und wählen Sie den Attributtyp für jedes Attribut aus. Verified Permissions verwendet die angegebenen Attributwerte, wenn Richtlinien anhand des Schemas überprüft werden. Wählen Sie aus, ob jedes Attribut erforderlich ist. Um ein Attribut zu entfernen, das der Aktion hinzugefügt wurde, wählen Sie neben dem Attribut die Option Entfernen aus.
16. Wählen Sie Aktion hinzufügen aus.
17. Nachdem alle Entitätstypen und Aktionen zum Schema hinzugefügt wurden, wählen Sie Änderungen speichern aus.

JSON mode

Während der Aktualisierung werden Sie feststellen, dass der JSON-Editor Ihren Code anhand der JSON-Syntax validiert und Fehler und Warnungen während der Bearbeitung identifiziert, sodass Sie Probleme schneller finden können. Darüber hinaus müssen Sie sich keine Gedanken über die Formatierung von JSON machen. Wählen Sie einfach Format JSON, sobald Sie Ihre Aktualisierungen vorgenommen haben, und das Format wird aktualisiert, sodass es der erwarteten JSON-Formatierung entspricht.

Um ein Schema für verifizierte Berechtigungen im JSON-Modus zu bearbeiten

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Schema aus.
3. Wählen Sie den JSON-Modus und dann Schema bearbeiten aus.

4. Geben Sie den Inhalt Ihres JSON-Schemas in das Feld Inhalt ein. Sie können Aktualisierungen Ihres Schemas erst speichern, wenn Sie alle Syntaxfehler behoben haben. Sie können Format JSON wählen, um die JSON-Syntax Ihres Schemas mit den empfohlenen Abständen und Einzügen zu formatieren.
5. Wählen Sie Änderungen speichern aus.

Aktivieren des Richtlinienvalidierungsmodus für Amazon Verified Permissions

Sie können den Richtlinienüberprüfungsmodus unter Verifizierte Berechtigungen festlegen, um zu steuern, ob Richtlinienänderungen anhand des Schemas in Ihrem Richtlinienspeicher validiert werden.

> **Important**
>
> Wenn Sie die Richtlinienüberprüfung aktivieren, werden alle Versuche, eine Richtlinie oder Richtlinienvorlage zu erstellen oder zu aktualisieren, anhand des Schemas im Richtlinienspeicher validiert. Verified Permissions lehnt den Anforderungsversuch ab, wenn die Überprüfung fehlschlägt. Aus diesem Grund empfehlen wir, die Validierung während der Entwicklung Ihrer Anwendung ausgeschaltet zu lassen und sie zu Testzwecken einzuschalten und sie eingeschaltet zu lassen, während Ihre Anwendung in Produktion ist.

AWS Management Console

So legen Sie den Richtlinienvalidierungsmodus für einen Richtlinienspeicher fest

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie Einstellungen aus.
3. Wählen Sie im Abschnitt Validierungsmodus für Richtlinien die Option Ändern aus.
4. Führen Sie eine der folgenden Aktionen aus:
   - Um die Richtlinienvalidierung zu aktivieren und zu erzwingen, dass alle Richtlinienänderungen anhand Ihres Schemas validiert werden müssen, wählen Sie das Optionsfeld Strikt (empfohlen).
   - Um die Richtlinienvalidierung für Richtlinienänderungen zu deaktivieren, wählen Sie das Optionsfeld Aus. Geben Sie ein, `confirm` um zu bestätigen, dass Aktualisierungen von Richtlinien nicht mehr anhand Ihres Schemas validiert werden.
5. Wählen Sie Änderungen speichern.

AWS CLI

Um den Validierungsmodus für einen Richtlinienspeicher festzulegen

Sie können den Validierungsmodus für einen Richtlinienspeicher ändern, indem Sie den UpdatePolicyStoreVorgang verwenden und einen anderen Wert für den ValidationSettingsParameter angeben.

```
$ aws verifiedpermissions update-policy-store \
    --validation-settings "mode=OFF",
    --policy-store-id PSEXAMPLEabcdefg111111
{
    "createdDate": "2023-05-17T18:36:10.134448+00:00",
    "lastUpdatedDate": "2023-05-17T18:36:10.134448+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "validationSettings": {
        "Mode": "OFF"
    }
}
```

Weitere Informationen finden Sie unter Policy Validation im Cedar Policy Language Reference Guide.

Richtlinien für verifizierte Berechtigungen von Amazon

Eine Richtlinie ist eine Erklärung, die einem Auftraggeber entweder erlaubt oder verbietet, eine oder mehrere Maßnahmen an einer Ressource durchzuführen. Jede Richtlinie wird unabhängig von allen anderen Richtlinien bewertet. Weitere Informationen darüber, wie die Richtlinien von Cedar strukturiert und bewertet werden, finden Sie unter Überprüfung der Cedar-Richtlinien anhand des Schemas im Referenzhandbuch zur Sprache der Cedar-Richtlinien.

Important

Wenn Sie Cedar-Richtlinien verfassen, die sich auf Prinzipale, Ressourcen und Aktionen beziehen, können Sie die eindeutigen Identifikatoren definieren, die für jedes dieser Elemente verwendet werden. Wir empfehlen Ihnen dringend, die folgenden Best Practices zu befolgen:

- Verwenden Sie universell eindeutige Bezeichner (UUIDs) für alle Haupt- und Ressourcen-Identifikatoren.

  Wenn beispielsweise ein Benutzer das Unternehmen `jane` verlässt und Sie später eine andere Person den Namen verwenden lassen, erhält dieser neue Benutzer automatisch Zugriff auf alles`jane`, was durch Richtlinien gewährt wird, auf die immer noch verwiesen wird. `User::"jane"` Cedar kann nicht zwischen dem neuen und dem alten Benutzer unterscheiden. Dies gilt sowohl für Prinzipal- als auch für Ressourcen-IDs. Verwenden Sie immer Identifikatoren, die garantiert einzigartig sind und niemals wiederverwendet werden, um sicherzustellen, dass Sie nicht versehentlich Zugriff gewähren, weil eine Richtlinie eine alte Kennung enthält.

  Wenn Sie eine UUID für eine Entität verwenden, empfehlen wir, dass Sie ihr die Kommentarspezifikation//und den „freundlichen“ Namen Ihrer Entität folgen. Dies trägt dazu bei, dass Ihre Richtlinien leichter verständlich sind. Zum Beispiel: principal == Role: :"a1b2c3d4-e5f6-a1b2-c3d4- „„//Administratoren EXAMPLE11111
- Nehmen Sie keine personenbezogenen, vertraulichen oder sensiblen Informationen als Teil der eindeutigen Kennung für Ihre Prinzipale oder Ressourcen auf. Diese Kennungen sind in Protokolleinträgen enthalten, die in Trails geteilt werden. AWS CloudTrail

Themen

- Statische Richtlinien für Amazon Verified Permissions erstellen

- Statische Richtlinien von Amazon Verified Permissions bearbeiten
- Kontext wird hinzugefügt
- Verwenden des Amazon Verified Permissions-Testbench
- Beispielrichtlinien für Amazon Verified Permissions

Statische Richtlinien für Amazon Verified Permissions erstellen

Sie können eine statische Richtlinie für Prinzipale erstellen, um ihnen zu erlauben oder zu verbieten, bestimmte Aktionen mit bestimmten Ressourcen für Ihre Anwendung auszuführen. Eine statische Richtlinie enthält spezifische Werte für `principal` und `resource` und kann bei Autorisierungsentscheidungen verwendet werden.

AWS Management Console

Um eine statische Richtlinie zu erstellen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).
3. Wählen Sie Richtlinie erstellen und dann Statische Richtlinie erstellen aus.

> **Note**
>
> Wenn Sie eine Richtlinienerklärung haben, die Sie verwenden möchten, fahren Sie mit Schritt 8 fort und fügen Sie die Richtlinie in den Abschnitt Richtlinie auf der nächsten Seite ein.

4. Wählen Sie im Abschnitt Auswirkung der Richtlinie aus, ob die Richtlinie zulassen oder verbieten soll, wenn eine Anfrage mit der Richtlinie übereinstimmt. Wenn Sie „Zulassen“ wählen, erlaubt die Richtlinie den Prinzipalen, die Aktionen mit den Ressourcen durchzuführen. Wenn Sie dagegen „Verbieten“ wählen, erlaubt die Richtlinie den Prinzipalen nicht, die Aktionen mit den Ressourcen durchzuführen.
5. Wählen Sie im Feld Principals Scope den Geltungsbereich der Principals aus, für die die Richtlinie gelten soll.

   - Wählen Sie Spezifischer Auftraggeber aus, um die Richtlinie auf einen bestimmten Prinzipal anzuwenden. Geben Sie den Entitätstyp und die Kennung für den Prinzipal an, der die in der Richtlinie angegebenen Aktionen ausführen darf oder nicht.

- Wählen Sie Gruppe von Prinzipalen aus, um die Richtlinie auf eine Gruppe von Prinzipalen anzuwenden. Geben Sie den Namen der Hauptgruppe in das Feld Gruppe von Prinzipalen ein.
- Wählen Sie Alle Prinzipale aus, um die Richtlinie auf alle Prinzipale in Ihrem Richtlinienspeicher anzuwenden.

6. Wählen Sie im Feld Umfang der Ressourcen den Umfang der Ressourcen aus, für die die Richtlinie gelten soll.

   - Wählen Sie Bestimmte Ressourcen aus, um die Richtlinie auf eine bestimmte Ressource anzuwenden. Geben Sie den Entitätstyp und die Kennung für die Ressource an, für die die Richtlinie gelten soll.
   - Wählen Sie Gruppe von Ressourcen aus, um die Richtlinie auf eine Gruppe von Ressourcen anzuwenden. Geben Sie den Namen der Ressourcengruppe in das Feld Ressourcengruppe ein.
   - Wählen Sie Alle Ressourcen aus, um die Richtlinie auf alle Ressourcen in Ihrem Richtlinienspeicher anzuwenden.

7. Wählen Sie im Abschnitt Geltungsbereich der Aktionen den Umfang der Ressourcen aus, für die die Richtlinie gelten soll.

   - Wählen Sie Spezifische Gruppe von Aktionen, um die Richtlinie auf eine Reihe von Aktionen anzuwenden. Aktivieren Sie die Kontrollkästchen neben den Aktionen, um die Richtlinie anzuwenden.
   - Wählen Sie Alle Aktionen aus, um die Richtlinie auf alle Aktionen in Ihrem Richtlinienspeicher anzuwenden.

8. Wählen Sie Weiter aus.
9. Überprüfen Sie im Abschnitt „Richtlinien“ Ihre Cedar-Richtlinie. Sie können Format wählen, um die Syntax Ihrer Richtlinie mit den empfohlenen Abständen und Einzügen zu formatieren. Weitere Informationen finden Sie im Cedar Policy Language Reference Guide unter Basic Policy Construction in Cedar.
10. Geben Sie im Abschnitt Details eine optionale Beschreibung der Richtlinie ein.
11. Wählen Sie Richtlinie erstellen aus.

AWS CLI

Um eine statische Richtlinie zu erstellen

Mithilfe des CreatePolicyVorgangs können Sie eine statische Richtlinie erstellen. Das folgende Beispiel erstellt eine einfache statische Richtlinie.

```
$ aws verifiedpermissions create-policy \
    --definition "{ \"static\": { \"Description\": \"MyTestPolicy\", \"Statement\":
 \"permit(principal,action,resource) when {principal.owner == resource.owner};\"}}"
 \
    --policy-store-id PSEXAMPLEabcdefg111111
{
"Arn": "arn:aws:verifiedpermissions::123456789012:policy/PSEXAMPLEabcdefg111111/
SPEXAMPLEabcdefg111111",
    "createdDate": "2023-05-16T20:33:01.730817+00:00",
    "lastUpdatedDate": "2023-05-16T20:33:01.730817+00:00",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "STATIC"
}
```

Statische Richtlinien von Amazon Verified Permissions bearbeiten

Sie können eine bestehende statische Richtlinie in Ihrem Richtlinienspeicher bearbeiten. Sie können statische Richtlinien nur direkt aktualisieren. Um eine mit einer Vorlage verknüpfte Richtlinie zu ändern, müssen Sie die Richtlinienvorlage aktualisieren. Weitere Informationen finden Sie unter Richtlinienvorlagen für Amazon Verified Permissions bearbeiten.

Sie können die folgenden Elemente einer statischen Richtlinie ändern:

- Die, `action` auf die in der Richtlinie verwiesen wird.
- Eine Bedingungsklausel wie `when` und`unless`.

Sie können die folgenden Elemente einer statischen Richtlinie nicht ändern. Um eines dieser Elemente zu ändern, müssen Sie die Richtlinie löschen und neu erstellen.

- Eine Richtlinie von einer statischen Richtlinie zu einer mit einer Vorlage verknüpften Richtlinie.
- Die Auswirkung einer statischen Richtlinie von `permit` oder. `forbid`
- Der, `principal` auf den eine statische Richtlinie verweist.
- Wird von einer statischen Richtlinie `resource` referenziert.

AWS Management Console

Um eine statische Richtlinie zu bearbeiten

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).
3. Wählen Sie das Optionsfeld neben der statischen Richtlinie, die Sie bearbeiten möchten, und wählen Sie dann Bearbeiten aus.
4. Aktualisieren Sie im Abschnitt Richtlinientext die Klausel `action` oder die Bedingungsklausel Ihrer statischen Richtlinie. Sie können den Richtlinieneffekt oder `principal` `resource` die Richtlinie nicht aktualisieren.
5. Wählen Sie Richtlinie aktualisieren.

> **Note**
>
> Wenn die Richtlinienüberprüfung im Richtlinienspeicher aktiviert ist, führt die Aktualisierung einer statischen Richtlinie dazu, dass Verified Permissions die Richtlinie anhand des Schemas im Richtlinienspeicher validiert. Wenn die aktualisierte statische Richtlinie die Validierung nicht besteht, schlägt der Vorgang fehl und das Update wird nicht gespeichert.

AWS CLI

Um eine statische Richtlinie zu bearbeiten

Sie können eine statische Richtlinie mithilfe des UpdatePolicyVorgangs bearbeiten. Im folgenden Beispiel wird eine einfache statische Richtlinie bearbeitet.

Im Beispiel wird die Datei verwendet`definition.txt`, um die Richtliniendefinition zu enthalten.

```
{
    "static": {
        "description":  "Grant everyone of janeFriends UserGroup access to the
 vacationFolder Album",
        "statement": "permit(principal in UserGroup::\"janeFriends\", action,
 resource in Album::\"vacationFolder\" );"
    }
}
```

Der folgende Befehl verweist auf diese Datei.

```
$ aws verifiedpermissions create-policy \
    --definition file://definition.txt \
    --policy-store-id PSEXAMPLEabcdefg111111

{
    "createdDate": "2023-06-12T20:33:37.382907+00:00",
    "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
        "entityId": "janeFriends",
        "entityType": "UserGroup"
    },
    "resource": {
        "entityId": "vacationFolder",
        "entityType": "Album"
    }
}
```

Kontext wird hinzugefügt

Kontext ist die Information, die für politische Entscheidungen relevant ist, aber nicht Teil der Identität Ihres Auftraggebers, Ihrer Aktion oder Ihrer Ressource ist. Der Anspruch auf Zugriffstoken ist Kontext. Möglicherweise möchten Sie eine Aktion nur von einer Reihe von Quell-IP-Adressen aus zulassen oder nur, wenn sich Ihr Benutzer mit MFA angemeldet hat. Ihre Anwendung hat Zugriff auf diese kontextbezogenen Sitzungsdaten und muss sie für Autorisierungsanfragen mit Daten füllen. Die Kontextdaten in einer Autorisierungsanfrage mit verifizierten Berechtigungen müssen in einem Element im JSON-Format vorliegen. `contextMap`

Die Beispiele, die diesen Inhalt veranschaulichen, stammen aus einem Beispielrichtlinienspeicher. Um dem nachzugehen, erstellen Sie den DigitalPetStoreBeispiel-Richtlinienspeicher in Ihrer Testumgebung.

Das folgende Kontextobjekt deklariert einen von jedem Cedar-Datentyp für eine Anwendung auf der Grundlage des DigitalPetStoreBeispielrichtlinienspeichers.

```
"context": {
```

```
  "contextMap": {
    "AccountCodes": {
      "set": [
        {
          "long": 111122223333
        },
        {
          "long": 444455556666
        },
        {
          "long": 123456789012
        }
      ]
    },
    "approvedBy": {
    "entityIdentifier": {
      "entityId": "Bob",
      "entityType": "DigitalPetStore::User"
    }
    },
    "MfaAuthorized": {
      "boolean": true
    },
    "NetworkInfo": {
      "record": {
        "IPAddress": {
          "string": "192.0.2.178"
        },
        "Country": {
          "string": "United States of America"
        },
        "SSL": {
          "boolean": true
        }
    }
    },
    "RequestedOrderCount": {
      "long": 4
    },
    "UserAgent": {
      "string": "My UserAgent 1.12"
    }
  }
```

```
}
```

Datentypen im Autorisierungskontext

Boolesch

Eine Binärdatei `true` oder ein `false` Wert. In diesem Beispiel `MfaAuthenticated` gibt der boolesche Wert von `true` for an, dass der Kunde eine Multi-Faktor-Authentifizierung durchgeführt hat, bevor er die Anzeige seiner Bestellung angefordert hat.

Einstellen

Eine Sammlung von Kontextelementen. Gruppenmitglieder können alle vom gleichen Typ sein, wie in diesem Beispiel, oder von unterschiedlichen Typen, einschließlich einer verschachtelten Menge. In dem Beispiel ist der Kunde mit 3 verschiedenen Konten verknüpft.

String

Eine Folge von Buchstaben, Zahlen oder Symbolen, die in '' Zeichen eingeschlossen sind. In diesem Beispiel steht die `UserAgent` Zeichenfolge für den Browser, mit dem der Kunde die Anzeige seiner Bestellung angefordert hat.

Long

Als ganze Zahl. Im Beispiel `RequestedOrderCount` gibt das an, dass diese Anfrage Teil eines Stapels ist, der darauf zurückzuführen ist, dass der Kunde darum gebeten hat, vier seiner vergangenen Bestellungen einzusehen.

Rekord

Eine Sammlung von Attributen. Sie müssen diese Attribute im Anforderungskontext deklarieren. Ein Richtlinienspeicher mit einem Schema muss diese Entität und die Attribute der Entität im Schema enthalten. In diesem Beispiel enthält der `NetworkInfo` Datensatz Informationen über die ursprüngliche IP-Adresse des Benutzers, die vom Client festgelegte Geolokalisierung dieser IP und die Verschlüsselung bei der Übertragung.

EntityIdentifier

Ein Verweis auf eine Entität und Attribute, die im `entities` Element der Anfrage deklariert sind. In dem Beispiel wurde die Bestellung des Benutzers vom Mitarbeiter genehmigtBob.

Um diesen Beispielkontext in der DigitalPetStoreBeispiel-App zu testen, müssen Sie Ihre Anfrage`entities`, Ihr Policy-Store-Schema und die statische Richtlinie mit der Beschreibung Customer Role — Get Order aktualisieren.

Ändern DigitalPetStore , um den Autorisierungskontext zu akzeptieren

Anfänglich DigitalPetStorehandelt es sich nicht um einen sehr komplexen Richtlinienspeicher. Er enthält keine vorkonfigurierten Richtlinien oder Kontextattribute zur Unterstützung des von uns vorgestellten Kontextes. Um ein Beispiel für eine Autorisierungsanfrage mit diesen Kontextinformationen auszuwerten, nehmen Sie die folgenden Änderungen an Ihrem Richtlinienspeicher und Ihrer Autorisierungsanfrage vor. Kontextbeispiele mit Zugriffstoken-Informationen als Kontext finden Sie unterZugriffstoken zuordnen.

Schema

Wenden Sie die folgenden Aktualisierungen auf Ihr Policy-Store-Schema an, um die neuen Kontext-Attribute zu unterstützen. Aktualisieren Sie `GetOrder actions` es wie folgt.

```
"GetOrder": {
  "memberOf": [],
  "appliesTo": {
    "resourceTypes": [
      "Order"
    ],
    "context": {
      "type": "Record",
      "attributes": {
        "AccountCodes": {
          "type": "Set",
          "required": true,
          "element": {
            "type": "Long"
          }
        },
        "approvedBy": {
          "name": "User",
          "required": true,
          "type": "Entity"
        },
        "MfaAuthorized": {
          "type": "Boolean",
          "required": true
```

```
        },
        "NetworkInfo": {
          "type": "NetworkInfo",
          "required": true
        },
        "RequestedOrderCount": {
          "type": "Long",
          "required": true
        },
        "UserAgent": {
          "required": true,
          "type": "String"
        }
      }
    },
    "principalTypes": [
      "User"
    ]
  }
}
```

Um auf den `NetworkInfo` in Ihrem Anforderungskontext genannten `record` Datentyp zu verweisen, erstellen Sie ein CommonType-Konstrukt in Ihrem Schema, indem Sie Ihrem Schema zuvor `actions` Folgendes hinzufügen. Ein `commonType` Konstrukt ist ein gemeinsam genutzter Satz von Attributen, den Sie auf verschiedene Entitäten anwenden können.

```
"commonTypes": {
  "NetworkInfo": {
    "attributes": {
      "IPAddress": {
        "type": "String",
        "required": true
      },
      "SSL": {
        "required": true,
        "type": "Boolean"
      },
      "Country": {
        "required": true,
        "type": "String"
      }
    },
    "type": "Record"
```

```
  }
},
```

Policy

Die folgende Richtlinie legt Bedingungen fest, die von jedem der bereitgestellten Kontextelemente erfüllt werden müssen. Sie baut auf der bestehenden statischen Richtlinie mit der Beschreibung Customer Role — Get Order auf. Diese Richtlinie verlangt zunächst nur, dass der Principal, der eine Anfrage stellt, der Eigentümer der Ressource ist.

```
permit (
    principal in DigitalPetStore::Role::"Customer",
    action in [DigitalPetStore::Action::"GetOrder"],
    resource
) when {
    principal == resource.owner &&
    context.AccountCodes.contains(111122223333) &&
    context.approvedBy in DigitalPetStore::Role::"Employee" &&
    context.MfaAuthorized == true &&
    context.NetworkInfo.Country like "*United States*" &&
    context.NetworkInfo.IPAddress like "192.0.2.*" &&
    context.NetworkInfo.SSL == true &&
    context.RequestedOrderCount <= 4 &&
    context.UserAgent like "*My UserAgent*"
};
```

Wir haben nun verlangt, dass die Anfrage zum Abrufen einer Bestellung die zusätzlichen Kontextbedingungen erfüllt, die wir der Anfrage hinzugefügt haben.

1. Der Benutzer muss sich mit MFA angemeldet haben.
2. Der Webbrowser des Benutzers `User-Agent` muss die Zeichenfolge `My UserAgent` enthalten.
3. Der Benutzer muss die Ansicht von 4 oder weniger Bestellungen angefordert haben.
4. Einer der Kontocodes des Benutzers muss sein`111122223333`.
5. Die IP-Adresse des Benutzers muss aus den Vereinigte Staaten stammen, er muss sich in einer verschlüsselten Sitzung befinden und seine IP-Adresse muss mit beginnen`192.0.2.`.
6. Ein Mitarbeiter muss seine Bestellung genehmigt haben. Im `entities` Element der Autorisierungsanfrage deklarieren wir einen BenutzerBob, der die Rolle von hat`Employee`.

Request body

Nachdem Sie Ihren Richtlinienspeicher mit dem entsprechenden Schema und der entsprechenden Richtlinie konfiguriert haben, können Sie diese Autorisierungsanfrage dem API-Vorgang „Verified Permissions“ vorlegen IsAuthorized. Beachten Sie, dass das `entities` Segment eine Definition von`Bob`, einem Benutzer mit der Rolle von, enthält`Employee`.

```
{
  "principal": {
    "entityType": "DigitalPetStore::User",
    "entityId": "Alice"
  },
  "action": {
    "actionType": "DigitalPetStore::Action",
    "actionId": "GetOrder"
  },
  "resource": {
    "entityType": "DigitalPetStore::Order",
    "entityId": "1234"
  },
  "context": {
    "contextMap": {
      "AccountCodes": {
        "set": [
          {"long": 111122223333},
          {"long": 444455556666},
          {"long": 123456789012}
        ]
      },
      "approvedBy": {
        "entityIdentifier": {
          "entityId": "Bob",
          "entityType": "DigitalPetStore::User"
        }
      },
      "MfaAuthorized": {
        "boolean": true
      },
      "NetworkInfo": {
        "record": {
          "Country": {"string": "United States of America"},
          "IPAddress": {"string": "192.0.2.178"},
          "SSL": {"boolean": true}
```

```
        }
      },
      "RequestedOrderCount":{
        "long": 4
      },
      "UserAgent": {
        "string": "My UserAgent 1.12"
      }
    }
  },
  "entities": {
    "entityList": [
      {
        "identifier": {
          "entityType": "DigitalPetStore::User",
          "entityId": "Alice"
        },
        "attributes": {
          "memberId": {
            "string": "801b87f2-1a5c-40b3-b580-eacad506d4e6"
          }
        },
        "parents": [
          {
            "entityType": "DigitalPetStore::Role",
            "entityId": "Customer"
          }
        ]
      },
      {
        "identifier": {
          "entityType": "DigitalPetStore::User",
          "entityId": "Bob"
        },
        "attributes": {
          "memberId": {
            "string": "49d9b81e-735d-429c-989d-93bec0bcfd8b"
          }
        },
        "parents": [
          {
            "entityType": "DigitalPetStore::Role",
            "entityId": "Employee"
          }
```

```
        ]
      },
      {
        "identifier": {
          "entityType": "DigitalPetStore::Order",
          "entityId": "1234"
        },
        "attributes": {
          "owner": {
            "entityIdentifier": {
              "entityType": "DigitalPetStore::User",
              "entityId": "Alice"
            }
          }
        },
        "parents": []
      }
     ]
   },
   "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

Verwenden des Amazon Verified Permissions-Testbench

Verwenden Sie den Prüfstand für verifizierte Berechtigungen, um Richtlinien für verifizierte Berechtigungen zu testen und Fehler zu beheben, indem Sie Autorisierungsanfragen für sie ausführen. Der Prüfstand verwendet die von Ihnen angegebenen Parameter, um zu ermitteln, ob die Cedar-Richtlinien in Ihrem Richtlinienspeicher die Anfrage autorisieren würden. Sie können beim Testen von Autorisierungsanfragen zwischen dem visuellen Modus und dem JSON-Modus wechseln. Weitere Informationen darüber, wie die Richtlinien von Cedar strukturiert und bewertet werden, finden Sie unter Basic Policy Construction in Cedar im Cedar Policy Language Reference Guide.

> **Note**
>
> Wenn Sie mithilfe von Verified Permissions eine Autorisierungsanfrage stellen, können Sie die Liste der Principals und Ressourcen als Teil der Anfrage im Abschnitt Zusätzliche Entitäten angeben. Sie können jedoch keine Details zu den Aktionen angeben. Sie müssen im Schema angegeben oder aus der Anfrage abgeleitet werden. Sie können keine Aktion in den Abschnitt Zusätzliche Entitäten einfügen.

Einen visuellen Überblick und eine Demonstration des Prüfstands finden Sie auf dem AWS YouTube Kanal unter Amazon Verified Permissions — Policy Creation and Testing (Primer Series #3).

Visual mode

> **Note**
>
> Sie müssen in Ihrem Richtlinienspeicher ein Schema definiert haben, um den visuellen Modus des Prüfstands verwenden zu können.

Um Richtlinien im visuellen Modus zu testen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite die Option Testbench aus.
3. Wählen Sie den Visuellen Modus.
4. Wählen Sie im Abschnitt Principal aus den Principaltypen in Ihrem Schema den Principal aus, der die Aktion ausführt. Geben Sie einen Bezeichner für den Principal in das Textfeld ein.
5. (Optional) Wählen Sie Übergeordnetes Element hinzufügen aus, um übergeordnete Entitäten für den angegebenen Prinzipal hinzuzufügen. Um ein übergeordnetes Objekt zu entfernen, das dem Prinzipal hinzugefügt wurde, wählen Sie neben dem Namen des übergeordneten Elements die Option Entfernen aus.
6. Geben Sie den Attributwert für jedes Attribut des angegebenen Prinzipals an. Der Prüfstand verwendet die angegebenen Attributwerte in der simulierten Autorisierungsanfrage.
7. Wählen Sie im Abschnitt Ressource die Ressource aus, auf die der Principal reagiert. Geben Sie einen Bezeichner für die Ressource in das Textfeld ein.
8. (Optional) Wählen Sie Übergeordnetes Element hinzufügen aus, um übergeordnete Entitäten für die angegebene Ressource hinzuzufügen. Um ein übergeordnetes Element zu entfernen, das der Ressource hinzugefügt wurde, wählen Sie neben dem Namen der übergeordneten Ressource die Option Entfernen aus.
9. Geben Sie den Attributwert für jedes Attribut der angegebenen Ressource an. Der Prüfstand verwendet die angegebenen Attributwerte in der simulierten Autorisierungsanfrage.
10. Wählen Sie im Abschnitt Aktion aus der Liste der gültigen Aktionen für den angegebenen Prinzipal und die angegebene Ressource die Aktion aus, die der Prinzipal ausführt.
11. Geben Sie den Attributwert für jedes Attribut der angegebenen Aktion an. Der Prüfstand verwendet die angegebenen Attributwerte in der simulierten Autorisierungsanfrage.

12. (Optional) Wählen Sie im Abschnitt Zusätzliche Entitäten die Option Entität hinzufügen aus, um Entitäten hinzuzufügen, die für die Autorisierungsentscheidung bewertet werden sollen.
13. Wählen Sie den Entitätsbezeichner aus der Dropdownliste aus und geben Sie den Entitätsbezeichner ein.
14. (Optional) Wählen Sie Übergeordnetes Element hinzufügen aus, um übergeordnete Entitäten für die angegebene Entität hinzuzufügen. Um ein übergeordnetes Element zu entfernen, das der Entität hinzugefügt wurde, wählen Sie neben dem Namen der übergeordneten Entität die Option Entfernen aus.
15. Geben Sie den Attributwert für jedes Attribut der angegebenen Entität an. Der Prüfstand verwendet die angegebenen Attributwerte in der simulierten Autorisierungsanfrage.
16. Wählen Sie Bestätigen, um die Entität dem Prüfstand hinzuzufügen.
17. Wählen Sie Autorisierungsanfrage ausführen, um die Autorisierungsanfrage für die Cedar-Richtlinien in Ihrem Richtlinienspeicher zu simulieren. Auf dem Prüfstand werden die Entscheidung, ob die Anfrage zugelassen oder abgelehnt wurde, zusammen mit Informationen zu den erfüllten Richtlinien oder den bei der Evaluierung aufgetretenen Fehlern angezeigt.

JSON mode

Um Richtlinien im JSON-Modus zu testen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite die Option Testbench aus.
3. Wählen Sie den JSON-Modus.
4. Wenn Sie im Abschnitt Anforderungsdetails ein Schema definiert haben, wählen Sie aus den Prinzipaltypen in Ihrem Schema den Principal aus, der die Aktion ausführt. Geben Sie einen Bezeichner für den Principal in das Textfeld ein.

   Wenn Sie kein Schema definiert haben, geben Sie den Principal in das Textfeld Principal ergreift Aktion ein.
5. Wenn Sie ein Schema definiert haben, wählen Sie die Ressource aus den Ressourcentypen in Ihrem Schema aus. Geben Sie einen Bezeichner für die Ressource in das Textfeld ein.

   Wenn Sie kein Schema definiert haben, geben Sie die Ressource in das Textfeld Ressource ein.

6. Wenn Sie ein Schema definiert haben, wählen Sie die Aktion aus der Liste der gültigen Aktionen für den angegebenen Prinzipal und die angegebene Ressource aus.

   Wenn Sie kein Schema definiert haben, geben Sie die Aktion in das Textfeld Aktion ein.
7. Geben Sie den Kontext der zu simulierenden Anforderung in das Feld Kontext ein. Der Anforderungskontext besteht aus zusätzlichen Informationen, die für Autorisierungsentscheidungen verwendet werden können.
8. Geben Sie im Feld Entitäten die Hierarchie der Entitäten und ihrer Attribute ein, die für die Autorisierungsentscheidung ausgewertet werden sollen.
9. Wählen Sie Autorisierungsanfrage ausführen, um die Autorisierungsanfrage für die Cedar-Richtlinien in Ihrem Richtlinienspeicher zu simulieren. Auf dem Prüfstand werden die Entscheidung, ob die Anfrage zugelassen oder abgelehnt wurde, zusammen mit Informationen zu den erfüllten Richtlinien oder den bei der Evaluierung aufgetretenen Fehlern angezeigt.

Beispielrichtlinien für Amazon Verified Permissions

Bei einigen der hier aufgeführten Richtlinienbeispiele handelt es sich um grundlegende Richtlinienbeispiele von Cedar, und andere beziehen sich auf verifizierte Berechtigungen. Die grundlegenden Links verweisen auf den Cedar Policy Language Reference Guide und sind dort enthalten. Weitere Informationen zur Cedar-Policy-Syntax finden Sie unter Basic Policy Construction in Cedar im Cedar Policy Language Reference Guide.

Beispiele für Richtlinien

- Ermöglicht den Zugriff auf einzelne Entitäten
- Ermöglicht den Zugriff auf Gruppen von Entitäten
- Ermöglicht den Zugriff für jede Entität
- Ermöglicht den Zugriff auf Attribute einer Entität (ABAC)
- Verweigert den Zugriff
- Verwendet die Klammernotation, um auf Token-Attribute zu verweisen
- Verwendet Punktnotation, um auf Attribute zu verweisen
- Spiegelt Amazon Cognito Cognito-ID-Token-Attribute wider
- Spiegelt die OIDC-ID-Token-Attribute wider
- Spiegelt die Attribute des Amazon Cognito Cognito-Zugriffstokens wider

- Spiegelt die Attribute von OIDC-Zugriffstoken wider

Verwendet die Klammernotation, um auf Token-Attribute zu verweisen

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die die Klammernotation verwendet, um auf Token-Attribute zu verweisen.

Weitere Informationen zur Verwendung von Tokenattributen in Richtlinien finden Sie unter Verifizierte BerechtigungenZuordnen von Identitätsanbieter-Tokens zum Schema.

```
permit (
    principal in MyCorp::UserGroup::"us-west-2_EXAMPLE|MyUserGroup",
    action,
    resource
) when {
    principal["cognito:username"] == "alice" &&
    principal["custom:employmentStoreCode"] == "petstore-dallas" &&
    principal has email && principal.email == "alice@example.com" &&
    context["ip-address"] like "192.0.2.*"
};
```

Verwendet Punktnotation, um auf Attribute zu verweisen

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die Punktnotation verwendet, um auf Attribute zu verweisen.

Weitere Informationen zur Verwendung von Tokenattributen in Richtlinien finden Sie unter Verifizierte BerechtigungenZuordnen von Identitätsanbieter-Tokens zum Schema.

```
permit(principal, action, resource)
when {
    principal.cognito.username == "alice" &&
    principal.custom.employmentStoreCode == "petstore-dallas" &&
    principal.tenant == "x11app-tenant-1" &&
    principal has email && principal.email == "alice@example.com"
};
```

Spiegelt Amazon Cognito Cognito-ID-Token-Attribute wider

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen könnten, die auf ID-Token-Attribute von Amazon Cognito verweist.

Weitere Informationen zur Verwendung von Token-Attributen in Richtlinien finden Sie unter Zuordnen von Identitätsanbieter-Tokens zum Schema Verifizierte Berechtigungen.

```
permit (
    principal in MyCorp::UserGroup::"us-west-2_EXAMPLE|MyUserGroup",
    action,
    resource
) when {
    principal["cognito:username"] == "alice" &&
    principal["custom:employmentStoreCode"] == "petstore-dallas" &&
    principal.tenant == "x11app-tenant-1" &&
    principal has email && principal.email == "alice@example.com"
};
```

Spiegelt die OIDC-ID-Token-Attribute wider

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen könnten, die auf ID-Token-Attribute eines OIDC-Anbieters verweist.

Weitere Informationen zur Verwendung von Tokenattributen in Richtlinien finden Sie unter Verifizierte Berechtigungen. Zuordnen von Identitätsanbieter-Tokens zum Schema

```
permit (
    principal in MyCorp::UserGroup::"MyOIDCProvider|MyUserGroup",
    action,
    resource
) when {
    principal.email_verified == true && principal.email == "alice@example.com" &&
    principal.phone_number_verified == true && principal.phone_number like "+1206*"
};
```

Spiegelt die Attribute des Amazon Cognito Cognito-Zugriffstokens wider

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die auf Zugriffstoken-Attribute von Amazon Cognito verweist.

Weitere Informationen zur Verwendung von Token-Attributen in Richtlinien finden Sie unter Zuordnen von Identitätsanbieter-Tokens zum Schema Verifizierte Berechtigungen.

```
permit(principal, action in [MyApplication::Action::"Read",
 MyApplication::Action::"GetStoreInventory"], resource)
```

```
when {
    context.token.client_id == "52n97d5afhfiu1c4di1k5m8f60" &&
    context.token.scope.contains("MyAPI/mydata.write")
};
```

Spiegelt die Attribute von OIDC-Zugriffstoken wider

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen könnten, die auf Zugriffstoken-Attribute eines OIDC-Anbieters verweist.

Weitere Informationen zur Verwendung von Tokenattributen in Richtlinien finden Sie unter Verifizierte Berechtigungen. Zuordnen von Identitätsanbieter-Tokens zum Schema

```
permit(
    principal,
    action in [MyApplication::Action::"Read",
 MyApplication::Action::"GetStoreInventory"],
    resource
)
when {
    context.token.client_id == "52n97d5afhfiu1c4di1k5m8f60" &&
    context.token.scope.contains("MyAPI-read")
};
```

Richtlinienvorlagen für Amazon Verified Permissions und Richtlinien, die mit Vorlagen verknüpft sind

Bei Verifizierten Berechtigungen handelt es sich bei Richtlinienvorlagen um Richtlinien mit Platzhaltern für `principalresource`, oder beide. Richtlinienvorlagen allein können nicht zur Bearbeitung von Autorisierungsanfragen verwendet werden. Um Autorisierungsanfragen bearbeiten zu können, muss auf der Grundlage einer Richtlinienvorlage eine mit einer Vorlage verknüpfte Richtlinie erstellt werden. Mit Richtlinienvorlagen kann eine Richtlinie einmal definiert und dann mit mehreren Prinzipalen und Ressourcen verwendet werden. Aktualisierungen der Richtlinienvorlage wirken sich auf alle Richtlinien aus, die die Vorlage verwenden. Weitere Informationen finden Sie unter Cedar-Richtlinienvorlagen im Cedar-Referenzhandbuch für Richtlinien in der Sprache.

Die folgende Richtlinienvorlage bietet beispielsweise `ReadEdit`, und `Comment` Berechtigungen für den Prinzipal und die Ressource, die die Richtlinienvorlage verwenden.

```
permit(
  principal == ?principal,
  action in [Action::"Read", Action::"Edit", Action::"Comment"],
  resource == ?resource
);
```

Wenn Sie eine Richtlinie erstellen würden, die auf dieser Vorlage `Editor` basiert, würde Ihre Anwendung, wenn ein Prinzipal als Bearbeiter für eine bestimmte Ressource bestimmt wird, eine Richtlinie erstellen, die dem Prinzipal Berechtigungen zum Lesen, Bearbeiten und Kommentieren der Ressource gewährt.

Im Gegensatz zu statischen Richtlinien sind mit Vorlagen verknüpfte Richtlinien dynamisch. Nehmen wir das vorherige Beispiel: Wenn Sie die `Comment` Aktion aus der Richtlinienvorlage entfernen würden, würden alle Richtlinien, die mit dieser Vorlage verknüpft sind oder auf ihr basieren, entsprechend aktualisiert und die in den Richtlinien angegebenen Prinzipale könnten keine Kommentare mehr zu den entsprechenden Ressourcen abgeben.

Weitere Beispiele für Richtlinien, die mit Vorlagen verknüpft sind, finden Sie unter. Beispiel für vorlagenverknüpfte Richtlinien mit Amazon Verified Permissions

Richtlinienvorlagen für Amazon Verified Permissions erstellen

Sie können Richtlinienvorlagen in Verified Permissions mit dem AWS Management Console AWS CLI, dem oder AWS erstellen SDKs. Mit Richtlinienvorlagen kann eine Richtlinie einmal definiert und dann mit mehreren Prinzipalen und Ressourcen verwendet werden. Sobald Sie eine Richtlinienvorlage erstellt haben, können Sie mit Vorlagen verknüpfte Richtlinien erstellen, um die Richtlinienvorlagen mit bestimmten Prinzipalen und Ressourcen zu verwenden. Weitere Informationen finden Sie unter Mit Vorlagen verknüpfte Richtlinien mit Amazon Verified Permissions erstellen.

AWS Management Console

Um eine Richtlinienvorlage zu erstellen

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Richtlinienvorlagen aus.
3. Wählen Sie Richtlinienvorlage erstellen aus.
4. Geben Sie im Abschnitt Details eine Beschreibung der Richtlinienvorlage ein.
5. Verwenden Sie im Abschnitt Text der Richtlinienvorlage Platzhalter?`principal`, damit Richtlinien, `?resource` die auf der Grundlage dieser Vorlage erstellt wurden, die von ihnen gewährten Berechtigungen anpassen können. Sie können Format wählen, um die Syntax Ihrer Richtlinienvorlage mit den empfohlenen Abständen und Einzügen zu formatieren.
6. Wählen Sie Richtlinienvorlage erstellen aus.

AWS CLI

Um eine Richtlinienvorlage zu erstellen

Mithilfe des CreatePolicyTemplateVorgangs können Sie eine Richtlinienvorlage erstellen. Im folgenden Beispiel wird eine Richtlinienvorlage mit einem Platzhalter für den Prinzipal erstellt.

Die Datei `template1.txt` enthält Folgendes.

```
"VacationAccess"
permit(
    principal in ?principal,
    action == Action::"view",
```

```
    resource == Photo::"VacationPhoto94.jpg"
);
```

```
$ aws verifiedpermissions create-policy-template \
    --description "Template for vacation picture access"
    --statement file://template1.txt
    --policy-store-id PSEXAMPLEabcdefg111111
{
    "createdDate": "2023-05-18T21:17:47.284268+00:00",
    "lastUpdatedDate": "2023-05-18T21:17:47.284268+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
}
```

Mit Vorlagen verknüpfte Richtlinien mit Amazon Verified Permissions erstellen

Sie können mit Vorlagen verknüpfte Richtlinien oder Richtlinien, die auf einer Richtlinienvorlage basieren, mit dem, oder dem AWS Management Console erstellen. AWS CLI AWS SDKs Mit Vorlagen verknüpfte Richtlinien bleiben mit ihren Richtlinienvorlagen verknüpft. Wenn Sie die Richtlinienerklärung in der Richtlinienvorlage ändern, verwenden alle mit dieser Vorlage verknüpften Richtlinien automatisch die neue Erklärung für alle Autorisierungsentscheidungen, die ab diesem Zeitpunkt getroffen werden.

Beispiele für Richtlinien, die mit Vorlagen verknüpft sind, finden Sie unter. Beispiel für vorlagenverknüpfte Richtlinien mit Amazon Verified Permissions

AWS Management Console

So erstellen Sie eine mit einer Vorlage verknüpfte Richtlinie, indem Sie eine Richtlinienvorlage instanziieren

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).
3. Wählen Sie Richtlinie erstellen und dann Richtlinie mit Vorlage erstellen aus.
4. Wählen Sie das Optionsfeld neben der Richtlinienvorlage, die Sie verwenden möchten, und klicken Sie dann auf Weiter.

5. Geben Sie den Principal und die Ressource ein, die für diese spezielle Instanz der mit der Vorlage verknüpften Richtlinie verwendet werden sollen. Die angegebenen Werte werden im Vorschaufeld für die Richtlinienerklärung angezeigt.

   > **Note**
   >
   > Die Werte Principal und Resource müssen dieselbe Formatierung wie statische Richtlinien haben. Um beispielsweise die `AdminUsers` Gruppe für den Prinzipal anzugeben, geben Sie ein`Group::"AdminUsers"`. Wenn Sie eingeben`AdminUsers`, wird ein Überprüfungsfehler angezeigt.

6. Wählen Sie Richtlinie mit Vorlagenverknüpfung erstellen aus.

   Die neue, mit der Vorlage verknüpfte Richtlinie wird unter Richtlinien angezeigt.

AWS CLI

Um eine mit einer Vorlage verknüpfte Richtlinie zu erstellen, indem Sie eine Richtlinienvorlage instanziieren

Sie können eine mit einer Vorlage verknüpfte Richtlinie erstellen, die auf eine vorhandene Richtlinienvorlage verweist und Werte für alle von der Vorlage verwendeten Platzhalter angibt.

Im folgenden Beispiel wird eine mit einer Vorlage verknüpfte Richtlinie erstellt, die eine Vorlage mit der folgenden Anweisung verwendet:

```
permit(
    principal in ?principal,
    action == PhotoFlash::Action::"view",
    resource == PhotoFlash::Photo::"VacationPhoto94.jpg"
);
```

Außerdem wird die folgende `definition.txt` Datei verwendet, um den Wert für den `definition` Parameter bereitzustellen:

```
{
    "templateLinked": {
        "policyTemplateId": "PTEXAMPLEabcdefg111111",
        "principal": {
            "entityType": "PhotoFlash::User",
```

```
            "entityId": "alice"
        }
    }
}
```

Die Ausgabe zeigt sowohl die Ressource, die sie aus der Vorlage bezieht, als auch den Prinzipal, den sie aus dem Definitionsparameter erhält

```
$ aws verifiedpermissions create-policy \
    --definition file://definition.txt
    --policy-store-id PSEXAMPLEabcdefg111111
{
    "createdDate": "2023-05-22T18:57:53.298278+00:00",
    "lastUpdatedDate": "2023-05-22T18:57:53.298278+00:00",
    "policyId": "TPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "TEMPLATELINKED",
    "principal": {
        "entityId": "alice",
        "entityType": "PhotoFlash::User"
    },
    "resource": {
        "entityId": "VacationPhoto94.jpg",
        "entityType": "PhotoFlash::Photo"
    }
}
```

Richtlinienvorlagen für Amazon Verified Permissions bearbeiten

Sie können Richtlinienvorlagen unter Verifizierte Berechtigungen mit dem AWS Management Console, dem oder dem AWS CLI bearbeiten oder aktualisieren AWS SDKs. Durch die Bearbeitung einer Richtlinienvorlage werden automatisch die Richtlinien aktualisiert, die mit der Vorlage verknüpft sind oder auf ihr basieren. Gehen Sie also beim Bearbeiten der Richtlinienvorlagen vorsichtig vor und stellen Sie sicher, dass Sie nicht versehentlich eine Änderung vornehmen, die Ihre Anwendung beschädigt.

Sie können die folgenden Elemente einer Richtlinienvorlage ändern:

- Die, `action` auf die in der Richtlinienvorlage verwiesen wird
- Eine Bedingungsklausel wie `when` und `unless`

Sie können die folgenden Elemente einer Richtlinienvorlage nicht ändern. Um eines dieser Elemente zu ändern, müssen Sie die Richtlinienvorlage löschen und neu erstellen.

- Die Wirkung einer Richtlinienvorlage von oder `permit` `forbid`
- Die, `principal` auf die in einer Richtlinienvorlage verwiesen wird
- Das, `resource` auf das in einer Richtlinienvorlage verwiesen wird

AWS Management Console

Um Ihre Richtlinienvorlagen zu bearbeiten

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Richtlinienvorlagen aus. In der Konsole werden alle Richtlinienvorlagen angezeigt, die Sie im aktuellen Richtlinienspeicher erstellt haben.
3. Wählen Sie das Optionsfeld neben einer Richtlinienvorlage, um Details zur Richtlinienvorlage anzuzeigen, z. B. wann die Richtlinienvorlage erstellt und aktualisiert wurde und welchen Inhalt sie hat.
4. Wählen Sie Bearbeiten, um Ihre Richtlinienvorlage zu bearbeiten. Aktualisieren Sie die Richtlinienbeschreibung und den Richtlinientext nach Bedarf und wählen Sie dann Richtlinienvorlage aktualisieren aus.
5. Sie können eine Richtlinienvorlage löschen, indem Sie auf das Optionsfeld neben einer Richtlinienvorlage klicken und dann Löschen wählen. Wählen Sie OK, um das Löschen der Richtlinienvorlage zu bestätigen.

AWS CLI

Um eine Richtlinienvorlage zu bearbeiten

Mithilfe des UpdatePolicyVorgangs können Sie eine statische Richtlinie erstellen. Im folgenden Beispiel wird die angegebene Richtlinienvorlage aktualisiert, indem der zugehörige Richtlinientext durch eine neue Richtlinie ersetzt wird, die in einer Datei definiert ist.

Inhalt der Datei`template1.txt`:

```
permit(
    principal in ?principal,
    action == Action::"view",
```

```
    resource in ?resource)
when {
    principal has department && principal.department == "research"
};
```

```
$ aws verifiedpermissions update-policy-template \
    --policy-template-id PTEXAMPLEabcdefg111111 \
    --description "My updated template description" \
    --statement file://template1.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
{
    "createdDate": "2023-05-17T18:58:48.795411+00:00",
    "lastUpdatedDate": "2023-05-17T19:18:48.870209+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
}
```

Beispiel für vorlagenverknüpfte Richtlinien mit Amazon Verified Permissions

Wenn Sie unter Verifizierte Berechtigungen mithilfe der Methode „Beispiel-Richtlinienspeicher“ einen Richtlinienspeicher erstellen, wird Ihr Richtlinienspeicher mit vordefinierten Richtlinien, Richtlinienvorlagen und einem Schema für das von Ihnen ausgewählte Beispielprojekt erstellt. Die folgenden Richtlinienbeispiele, die mit Vorlagen für verifizierte Berechtigungen verknüpft sind, können mit den Beispielrichtlinienspeichern und ihren jeweiligen Richtlinien, Richtlinienvorlagen und Schemas verwendet werden.

Beispiele für PhotoFlash

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen können, die die Richtlinienvorlage „Beschränkten Zugriff auf nicht private geteilte Fotos mit einem einzelnen Benutzer und Foto gewähren“ verwendet.

> **Note**
>
> In der Sprache der Richtlinien von Cedar wird eine Entität als sie selbst betrachtet. Daher `principal in User::"Alice"` ist gleichbedeutend mit `principal == User::"Alice"`.

```
permit (
 principal in PhotoFlash::User::"Alice",
 action in PhotoFlash::Action::"SharePhotoLimitedAccess",
 resource in PhotoFlash::Photo::"VacationPhoto94.jpg"
 );
```

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen könnten, die die Richtlinienvorlage „Beschränkten Zugriff auf nicht private, mit einem einzelnen Benutzer und Album geteilte Fotos gewähren“ verwendet.

```
permit (
 principal in PhotoFlash::User::"Alice",
 action in PhotoFlash::Action::"SharePhotoLimitedAccess",
 resource in PhotoFlash::Album::"Italy2023"
 );
```

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen könnten, die die Richtlinienvorlage Beschränkten Zugriff auf nicht private geteilte Fotos mit einer Freundesgruppe und einem einzelnen Foto verwendet.

```
permit (
 principal in PhotoFlash::FriendGroup::"Jane::MySchoolFriends",
 action in PhotoFlash::Action::"SharePhotoLimitedAccess",
 resource in PhotoFlash::Photo::"VacationPhoto94.jpg"
 );
```

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen könnten, die die Richtlinienvorlage Beschränkten Zugriff auf nicht private geteilte Fotos mit einer Freundesgruppe und einem Album verwendet.

```
permit (
 principal in PhotoFlash::FriendGroup::"Jane::MySchoolFriends",
 action in PhotoFlash::Action::"SharePhotoLimitedAccess",
 resource in PhotoFlash::Album::"Italy2023"
 );
```

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen könnten, bei der die Richtlinienvorlage Vollzugriff auf nicht private geteilte Fotos mit einer Freundesgruppe und einem einzelnen Foto verwendet wird.

```
permit (
 principal in PhotoFlash::UserGroup::"Jane::MySchoolFriends",
 action in PhotoFlash::Action::"SharePhotoFullAccess",
 resource in PhotoFlash::Photo::"VacationPhoto94.jpg"
 );
```

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen könnten, die die Richtlinienvorlage Benutzer von einem Konto blockieren verwendet.

```
forbid(
 principal == PhotoFlash::User::"Bob",
 action,
 resource in PhotoFlash::Account::"Alice-account"
 );
```

DigitalPetStore Beispiele

Der DigitalPetStore Beispielrichtlinienspeicher enthält keine Richtlinienvorlagen. Sie können die im Richtlinienspeicher enthaltenen Richtlinien anzeigen, indem Sie nach dem Erstellen des DigitalPetStoreBeispielrichtlinienspeichers im Navigationsbereich auf der linken Seite Richtlinien auswählen.

Beispiele für TinyToDo

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen können, die die Richtlinienvorlage verwendet, die dem Betrachter Zugriff auf einzelne Benutzer und Aufgabenlisten gewährt.

```
permit (
    principal == TinyTodo::User::"https://cognito-idp.us-east-1.amazonaws.com/us-
east-1_h2aKCU1ts|5ae0c4b1-6de8-4dff-b52e-158188686f31|bob",
    action in [TinyTodo::Action::"ReadList", TinyTodo::Action::"ListTasks"],
    resource == TinyTodo::List::"1"
);
```

Das folgende Beispiel zeigt, wie Sie eine mit einer Vorlage verknüpfte Richtlinie erstellen könnten, die die Richtlinienvorlage verwendet, die dem Editor Zugriff auf einen einzelnen Benutzer und eine Aufgabenliste gewährt.

```
permit (
```

```
    principal == TinyTodo::User::"https://cognito-idp.us-east-1.amazonaws.com/us-
east-1_h2aKCU1ts|5ae0c4b1-6de8-4dff-b52e-158188686f31|bob",
    action in [
        TinyTodo::Action::"ReadList",
        TinyTodo::Action::"UpdateList",
        TinyTodo::Action::"ListTasks",
        TinyTodo::Action::"CreateTask",
        TinyTodo::Action::"UpdateTask",
        TinyTodo::Action::"DeleteTask"
    ],
    resource == TinyTodo::List::"1"
);
```

Verwenden von Amazon Verified Permissions mit Identitätsanbietern

Eine Identitätsquelle ist eine Darstellung eines externen Identitätsanbieters (IdP) in Amazon Verified Permissions. Identitätsquellen stellen Informationen von einem Benutzer bereit, der sich bei einem IdP authentifiziert hat, der eine Vertrauensbeziehung zu Ihrem Richtlinienspeicher unterhält. Wenn Ihre Anwendung eine Autorisierungsanfrage mit einem Token aus einer Identitätsquelle stellt, kann Ihr Richtlinienspeicher anhand von Benutzereigenschaften und Zugriffsberechtigungen Autorisierungsentscheidungen treffen. Sie können einen Amazon Cognito Cognito-Benutzerpool oder einen benutzerdefinierten OpenID Connect (OIDC) IdP als Identitätsquelle hinzufügen.

Sie können OpenID Connect (OIDC) -Identitätsanbieter (IdPs) mit verifizierten Berechtigungen verwenden. Ihre Anwendung kann Autorisierungsanfragen mit JSON-Webtoken (JWTs) generieren, die von einem OIDC-konformen Identitätsanbieter generiert wurden. Die Benutzeridentität im Token ist der Prinzipal-ID zugeordnet. Bei ID-Tokens ordnet Verified Permissions Attributansprüche den Hauptattributen zu. Mit Zugriffstoken werden diese Ansprüche dem Kontext zugeordnet. Mit beiden Tokentypen können Sie einen Anspruch quasi einer Prinzipalgruppe `groups` zuordnen und Richtlinien erstellen, die die rollenbasierte Zugriffskontrolle (RBAC) bewerten.

> **Note**
>
> Verified Permissions trifft Autorisierungsentscheidungen auf der Grundlage von Informationen aus einem IdP-Token, interagiert jedoch in keiner Weise direkt mit dem IdP.

Eine step-by-step exemplarische Vorgehensweise, die die Autorisierungslogik für Amazon API Gateway REST APIs mithilfe eines Amazon Cognito-Benutzerpools oder eines OIDC-Identitätsanbieters erstellt, finden Sie unter Autorisieren von API-Gateways APIs mithilfe von Amazon Verified Permissions with Amazon Cognito oder Bring Your Own Identity Provider im Security Blog.AWS

Themen

- Arbeiten mit Amazon Cognito Cognito-Identitätsquellen
- Arbeiten mit OIDC-Identitätsquellen
- Validierung von Kunden und Zielgruppen

- Kundenseitige Autorisierung für JWTs
- Identitätsquellen für Amazon Verified Permissions erstellen
- Identitätsquellen für Amazon Verified Permissions bearbeiten
- Zuordnen von Identitätsanbieter-Tokens zum Schema

Arbeiten mit Amazon Cognito Cognito-Identitätsquellen

Verified Permissions arbeitet eng mit Amazon Cognito Cognito-Benutzerpools zusammen. Amazon Cognito JWTs hat eine vorhersehbare Struktur. Verified Permissions erkennt diese Struktur und zieht den größtmöglichen Nutzen aus den darin enthaltenen Informationen. Sie können beispielsweise ein Autorisierungsmodell für die rollenbasierte Zugriffskontrolle (RBAC) implementieren, das entweder ID-Token oder Zugriffstoken verwendet.

Die Identitätsquelle eines neuen Amazon Cognito Cognito-Benutzerpools benötigt die folgenden Informationen:

- Das AWS-Region.
- Die Benutzerpool-ID.
- Der Haupt-Entitätstyp, den Sie Ihrer Identitätsquelle zuordnen möchten, zum Beispiel`MyCorp::User`.
- Der Entitätstyp der Prinzipalgruppe, den Sie Ihrer Identitätsquelle zuordnen möchten`MyCorp::UserGroup`.
- Der Client IDs aus Ihrem Benutzerpool, den Sie autorisieren möchten, Anfragen an Ihren Richtlinienspeicher zu stellen.

Da Verified Permissions nur mit Amazon Cognito Cognito-Benutzerpools in demselben funktioniert AWS-Konto, können Sie keine Identitätsquelle in einem anderen Konto angeben. Verified Permissions legt beispielsweise das Entitätspräfix — die Identitätsquellen-ID, auf die Sie in Richtlinien verweisen müssen, die sich auf Benutzerpool-Prinzipale beziehen — auf die ID Ihres Benutzerpools fest. `us-west-2_EXAMPLE` In diesem Fall würden Sie auf einen Benutzer in diesem Benutzerpool mit der ID als verweisen `a1b2c3d4-5678-90ab-cdef-EXAMPLE22222` `us-west-2_EXAMPLE|a1b2c3d4-5678-90ab-cdef-EXAMPLE22222`

Ansprüche auf Benutzerpool-Tokens können Attribute, Bereiche, Gruppen IDs, Client- und benutzerdefinierte Daten enthalten. Amazon Cognito JWTs kann eine Vielzahl von Informationen, die

zu Autorisierungsentscheidungen beitragen können, in Verifizierte Berechtigungen aufnehmen. Dazu zählen:

1. Nutzername und Gruppenansprüche mit einem Präfix `cognito:`
2. Benutzerdefinierte Benutzerattribute mit einem `custom: prefix`
3. Zur Laufzeit hinzugefügte benutzerdefinierte Ansprüche
4. OIDC-Standardansprüche wie und `sub` `email`

Wir behandeln diese Ansprüche ausführlich und erfahren, wie sie verwaltet werden, in den Richtlinien für verifizierte Berechtigungen, unter. Zuordnen von Identitätsanbieter-Tokens zum Schema

> **Important**
>
> Sie können Amazon Cognito Cognito-Token zwar vor ihrem Ablauf widerrufen, sie JWTs gelten jedoch als zustandslose Ressourcen, die eigenständig sind und über eine Signatur und Gültigkeit verfügen. Von Diensten, die dem JSON Web Token RFC 7519 entsprechen, wird erwartet, dass sie Token aus der Ferne validieren und müssen sie nicht beim Emittenten validieren. Das bedeutet, dass es verifizierten Berechtigungen möglich ist, Zugriff auf der Grundlage eines Tokens zu gewähren, das für einen Benutzer gesperrt oder ausgestellt und später gelöscht wurde. Um dieses Risiko zu minimieren, empfehlen wir Ihnen, Ihre Token mit der kürzest möglichen Gültigkeitsdauer zu erstellen und Aktualisierungstoken zu widerrufen, wenn Sie die Autorisierung zur Fortsetzung der Sitzung eines Benutzers entfernen möchten. Weitere Informationen finden Sie unter Benutzersitzungen mit Token-Widerruf beenden

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen könnten, die auf einige Ansprüche der Amazon Cognito Cognito-Benutzerpools verweist, die mit einem Prinzipal verknüpft sind.

```
permit(
     principal,
     action,
     resource == ExampleCo::Photo::"VacationPhoto94.jpg"
)
when {
     principal["cognito:username"]) == "alice" &&
     principal["custom:department"]) == "Finance"
};
```

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die auf einen Prinzipal verweist, der ein Benutzer in einem Cognito-Benutzerpool ist. Beachten Sie, dass die Prinzipal-ID die Form von `"<userpool-id>|<sub>"` hat.

```
permit(
     principal == ExampleCo::User::"us-east-1_example|a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
     action,
     resource == ExampleCo::Photo::"VacationPhoto94.jpg"
);
```

Die Richtlinien von Cedar für Identitätsquellen in Benutzerpools in Verified Permissions verwenden eine spezielle Syntax für Anspruchsnamen, die andere Zeichen als alphanumerische Zeichen und Unterstriche () _ enthalten. Dazu gehören auch Ansprüche auf Benutzerpool-Präfixe, die ein `:` Zeichen wie `cognito:username` und enthalten. `custom:department` Um eine Richtlinienbedingung zu schreiben, die auf den `custom:department` Anspruch `cognito:username` oder verweist, schreiben Sie sie jeweils als `principal["cognito:username"]` und`principal["custom:department"]`.

> **Note**
>
> Wenn ein Token einen Anspruch mit dem `custom:` Präfix `cognito:` oder und einen Anspruchsnamen mit dem wörtlichen Wert `cognito` oder enthält`custom`, schlägt eine Autorisierungsanfrage mit a IsAuthorizedWithToken`ValidationException`fehl.

Weitere Informationen zur Zuordnung von Ansprüchen finden Sie unterZuordnen von ID-Token zum Schema. Weitere Informationen zur Autorisierung für Amazon Cognito-Benutzer finden Sie unter Autorisierung mit von Amazon verifizierten Berechtigungen im Amazon Cognito Developer Guide.

Arbeiten mit OIDC-Identitätsquellen

Sie können auch jeden kompatiblen OpenID Connect (OIDC) IdP als Identitätsquelle für einen Richtlinienspeicher konfigurieren. OIDC-Anbieter ähneln Amazon Cognito Cognito-Benutzerpools: Sie produzieren JWTs als Produkt der Authentifizierung. Um einen OIDC-Anbieter hinzuzufügen, müssen Sie eine Aussteller-URL angeben

Für eine neue OIDC-Identitätsquelle sind die folgenden Informationen erforderlich:

- Die URL des Ausstellers. Verifizierte Berechtigungen müssen in der Lage sein, einen `.well-known/openid-configuration` Endpunkt unter dieser URL zu erkennen.
- CNAME-Einträge, die keine Platzhalter enthalten. `a.example.com`Kann beispielsweise nicht zugeordnet werden. `*.example.net` Umgekehrt `*.example.com` kann nicht zugeordnet werden. `a.example.net`
- Der Tokentyp, den Sie in Autorisierungsanfragen verwenden möchten. In diesem Fall haben Sie Identitätstoken gewählt.
- Der Benutzer-Entitätstyp, den Sie Ihrer Identitätsquelle zuordnen möchten, zum Beispiel`MyCorp::User`.
- Zum Beispiel der Gruppen-Entitätstyp, den Sie Ihrer Identitätsquelle zuordnen möchten`MyCorp::UserGroup`.
- Ein Beispiel für ein ID-Token oder eine Definition der Ansprüche im ID-Token.
- Das Präfix, das Sie auf die Benutzer- und Gruppenentität anwenden möchten IDs. In der CLI und API können Sie dieses Präfix wählen. In Richtlinienspeichern, die Sie mit der Option „Mit API Gateway und einem Identitätsanbieter einrichten“ oder „Geführte Einrichtung“ erstellen, weist Verified Permissions beispielsweise ein Präfix mit dem Namen des Ausstellers minus `https://` zu. `MyCorp::User::"auth.example.com|a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"`

Weitere Informationen zur Verwendung von API-Vorgängen zur Autorisierung von Anfragen aus OIDC-Quellen finden Sie unter. Verfügbare API-Operationen für die Autorisierung

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen könnten, die Mitarbeitern der Buchhaltungsabteilung Zugriff auf Jahresabschlussberichte gewährt, die vertraulich eingestuft sind und sich nicht in einem Außenbüro befinden. Verified Permissions leitet diese Attribute aus den Ansprüchen im ID-Token des Prinzipals ab.

Beachten Sie, dass Sie beim Verweisen auf eine Gruppe im Prinzipal den `in` Operator verwenden müssen, damit die Richtlinie korrekt ausgewertet wird.

```
permit(
     principal in MyCorp::UserGroup::"MyOIDCProvider|Accounting",
     action,
     resource in MyCorp::Folder::"YearEnd2024"
) when {
     principal.jobClassification == "Confidential" &&
     !(principal.location like "SatelliteOffice*")
};
```

Validierung von Kunden und Zielgruppen

Wenn Sie einem Richtlinienspeicher eine Identitätsquelle hinzufügen, verfügt Verified Permissions über Konfigurationsoptionen, mit denen überprüft wird, ob ID und Zugriffstoken wie vorgesehen verwendet werden. Diese Überprüfung erfolgt bei der Verarbeitung von `IsAuthorizedWithToken` und `BatchIsAuthorizedWithToken` API-Anfragen. Das Verhalten unterscheidet sich zwischen ID- und Zugriffstoken sowie zwischen Amazon Cognito- und OIDC-Identitätsquellen. Bei Anbietern von Amazon Cognito Cognito-Benutzerpools kann Verified Permissions die Client-ID sowohl in ID- als auch in Zugriffstoken validieren. Bei OIDC-Anbietern kann Verified Permissions die Client-ID in ID-Token und die Zielgruppe in Zugriffstoken validieren.

Eine Client-ID ist eine Kennung, die beispielsweise der Identitätsanbieter-Instanz zugeordnet ist, die Ihre Anwendung verwendet. `1example23456789` Eine Zielgruppe ist beispielsweise ein URL-Pfad, der der vertrauenden Partei oder dem Ziel des Zugriffstokens zugeordnet ist`https://mytoken.example.com`. Bei der Verwendung von Zugriffstoken ist der `aud` Anspruch immer mit der Zielgruppe verknüpft.

Verified Permissions führt die Überprüfung der Identitätsquelle, der Zielgruppe und des Clients wie folgt durch:

Amazon Cognito

- Amazon Cognito Cognito-ID-Token haben einen `aud` Anspruch, der die App-Client-ID enthält. Zugriffstoken haben einen `client_id` Anspruch, der auch die App-Client-ID enthält.

  Wenn Sie in Ihrer Identitätsquelle einen oder mehrere Werte für die Validierung von Client-Anwendungen eingeben, vergleicht Verified Permissions diese Liste von App-Clients IDs mit dem `aud` ID-Token-Anspruch oder dem `client_id` Zugriffstoken-Anspruch. Verified Permissions validiert keine Zielgruppen-URL einer vertrauenden Partei für Amazon Cognito Cognito-Identitätsquellen.

OIDC

- OIDC-ID-Tokens haben einen `aud` Anspruch, der den Kunden enthält, wie z. B. IDs `1example23456789`

  OIDC-Zugriffstoken haben einen `aud` Anspruch, der die Zielgruppen-URL für das Token enthält, wie z. B., und einen `client_id` Anspruch`https://myapplication.example.com`, der den Client IDs enthält, z. B. `1example23456789`

Geben Sie bei der Einrichtung Ihres Richtlinienspeichers einen oder mehrere Werte für die Zielgruppenvalidierung ein, die Ihr Richtlinienspeicher verwendet, um die Zielgruppe eines Tokens zu validieren.

- ID-Tokens — Verified Permissions validiert die Kunden-ID, indem überprüft wird, ob mindestens ein Mitglied des Kunden IDs im `aud` Antrag einem Wert für die Zielgruppenvalidierung entspricht.
- Zugriffstoken — Verifizierte Berechtigungen validieren die Zielgruppe, indem überprüft wird, ob die URL im `aud` Anspruch mit einem Zielgruppenvalidierungswert übereinstimmt. Wenn kein `aud` Anspruch besteht, kann die Zielgruppe anhand der `client_id` Ansprüche `cid` oder bestätigt werden. Erkundigen Sie sich bei Ihrem Identitätsanbieter nach der korrekten Angabe und dem richtigen Format für die Zielgruppe.

Kundenseitige Autorisierung für JWTs

Möglicherweise möchten Sie JSON-Webtoken in Ihrer Anwendung verarbeiten und deren Ansprüche an Verified Permissions weiterleiten, ohne eine Identitätsquelle für den Richtlinienspeicher zu verwenden. Sie können Ihre Entitätsattribute aus einem JSON-Webtoken (JWT) extrahieren und in verifizierte Berechtigungen umwandeln.

Dieses Beispiel zeigt, wie Sie verifizierte Berechtigungen von einer Anwendung aus aufrufen könnten, die ein JWT verwendet[1].

```
async function authorizeUsingJwtToken(jwtToken) {

    const payload = await verifier.verify(jwtToken);

    let principalEntity = {
        entityType: "PhotoFlash::User", // the application needs to fill in the
 relevant user type
        entityId: payload["sub"], // the application need to use the claim that
 represents the user-id
    };
    let resourceEntity = {
        entityType: "PhotoFlash::Photo", //the application needs to fill in the
 relevant resource type
        entityId: "jane_photo_123.jpg", // the application needs to fill in the
 relevant resource id
    };
    let action = {
```

```
        actionType: "PhotoFlash::Action", //the application needs to fill in the
 relevant action id
        actionId: "GetPhoto", //the application needs to fill in the relevant action
 type
    };
    let entities = {
        entityList: [],
    };
    entities.entityList.push(...getUserEntitiesFromToken(payload));
    let policyStoreId = "PSEXAMPLEabcdefg111111"; // set your own policy store id

    const authResult = await client
        .isAuthorized({
        policyStoreId: policyStoreId,
        principal: principalEntity,
        resource: resourceEntity,
        action: action,
        entities,
        })
        .promise();

    return authResult;

}

function getUserEntitiesFromToken(payload) {
  let attributes = {};
  let claimsNotPassedInEntities = ['aud', 'sub', 'exp', 'jti', 'iss'];
  Object.entries(payload).forEach(([key, value]) => {
    if (claimsNotPassedInEntities.includes(key)) {
        return;
    }
    if (Array.isArray(value)) {
      var attibuteItem = [];
      value.forEach((item) => {
        attibuteItem.push({
          string: item,
        });
      });
      attributes[key] = {
        set: attibuteItem,
      };
    } else if (typeof value === 'string') {
      attributes[key] = {
```

```
        string: value,
      }
    } else if (typeof value === 'bigint' || typeof value ==='number') {
        attributes[key] = {
            long: value,
          }
    } else if (typeof value === 'boolean') {
        attributes[key] = {
            boolean: value,
       }
    }

  });

  let entityItem = {
    attributes: attributes,
    identifier: {
      entityType: "PhotoFlash::User",
      entityId: payload["sub"], // the application needs to use the claim that
 represents the user-id
    }
  };
  return [entityItem];
}
```

[1] In diesem Codebeispiel wird die aws-jwt-verifyBibliothek zur Überprüfung JWTs verwendet, die mit OIDC-kompatibel signiert wurde. IdPs

Identitätsquellen für Amazon Verified Permissions erstellen

Das folgende Verfahren fügt einem vorhandenen Richtlinienspeicher eine Identitätsquelle hinzu. Nachdem Sie Ihre Identitätsquelle hinzugefügt haben, müssen Sie Ihrem Schema Attribute hinzufügen.

Sie können auch eine Identitätsquelle erstellen, wenn Sie in der Konsole „Verifizierte Berechtigungen“ einen neuen Richtlinienspeicher erstellen. In diesem Prozess können Sie die Ansprüche in Ihren Identitätsquellen-Token automatisch in Entitätsattribute importieren. Wählen Sie die Option Geführte Einrichtung oder Einrichtung mit API Gateway und einem Identitätsanbieter. Mit diesen Optionen werden auch erste Richtlinien erstellt.

> **Note**
>
> Identitätsquellen sind im Navigationsbereich auf der linken Seite erst verfügbar, wenn Sie einen Richtlinienspeicher erstellt haben. Identitätsquellen, die Sie erstellen, sind dem aktuellen Richtlinienspeicher zugeordnet.

Sie können den Hauptentitätstyp weglassen, wenn Sie eine Identitätsquelle mit create-identity-sourcein der AWS CLI oder CreateIdentitySourcein der Verified Permissions API erstellen. Ein leerer Entitätstyp erstellt jedoch eine Identitätsquelle mit dem Entitätstyp`AWS::Cognito`. Dieser Entitätsname ist nicht mit dem Richtlinienspeicherschema kompatibel. Um Amazon Cognito Cognito-Identitäten in Ihr Policy Store-Schema zu integrieren, müssen Sie den Prinzipal-Entitätstyp auf eine unterstützte Policy Store-Entität festlegen.

Themen

- Amazon Cognito Cognito-Identitätsquelle
- OIDC-Identitätsquelle

Amazon Cognito Cognito-Identitätsquelle

AWS Management Console

So erstellen Sie eine Identitätsquelle für Amazon Cognito Cognito-Benutzerpools

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Identitätsquellen aus.
3. Wählen Sie Identitätsquelle erstellen aus.
4. Wählen Sie in den Cognito-Benutzerpooldetails die Benutzerpool-ID für Ihre Identitätsquelle aus AWS-Region und geben Sie sie ein.
5. Wählen Sie in der Prinzipalkonfiguration für Principal Type den Entitätstyp für Principals aus dieser Quelle aus. Identitäten aus den verbundenen Amazon Cognito Cognito-Benutzerpools werden dem ausgewählten Prinzipaltyp zugeordnet.
6. Wählen Sie in der Gruppenkonfiguration die Option Cognito-Gruppe verwenden aus, wenn Sie den `cognito:groups` Benutzerpoolanspruch zuordnen möchten. Wählen Sie einen Entitätstyp, der dem Prinzipaltyp übergeordnet ist.

7. Wählen Sie unter Validierung der Client-Anwendung aus, ob die Client-Anwendung validiert werden soll IDs.

   - Um die Client-Anwendung zu validieren IDs, wählen Sie Nur Tokens mit passender Client-Anwendung akzeptieren IDs. Wählen Sie Neue Client-Anwendungs-ID hinzufügen für jede zu validierende Client-Anwendungs-ID aus. Um eine hinzugefügte Client-Anwendungs-ID zu entfernen, klicken Sie neben der Client-Anwendungs-ID auf Entfernen.
   - Wählen Sie Client-Anwendung nicht validieren IDs, wenn Sie die Client-Anwendung nicht validieren möchten IDs.

8. Wählen Sie Identitätsquelle erstellen.

Wenn Ihr Richtlinienspeicher über ein Schema verfügt, müssen Sie, bevor Sie in Ihren Cedar-Richtlinien auf Attribute verweisen können, die Sie aus Identitäts- oder Zugriffstoken extrahieren, Ihr Schema aktualisieren, damit Cedar weiß, welche Art von Prinzipal Ihre Identitätsquelle erstellt. Diese Ergänzung zum Schema muss die Attribute enthalten, auf die Sie in Ihren Cedar-Richtlinien verweisen möchten. Weitere Informationen zur Zuordnung von Amazon Cognito-Tokenattributen zu Cedar-Hauptattributen finden Sie unterZuordnen von Identitätsanbieter-Tokens zum Schema.

Wenn Sie einen API-verknüpften Richtlinienspeicher erstellen oder beim Erstellen von Richtlinienspeichern die Option Setup with API Gateway und Identity Provider verwenden, fragt Verified Permissions Ihren Benutzerpool nach Benutzerattributen ab und erstellt ein Schema, in dem Ihr Prinzipaltyp mit Benutzerpool-Attributen gefüllt wird.

AWS CLI

So erstellen Sie eine Identitätsquelle für Amazon Cognito Cognito-Benutzerpools

Sie können mithilfe des CreateIdentitySourceVorgangs eine Identitätsquelle erstellen. Im folgenden Beispiel wird eine Identitätsquelle erstellt, die auf authentifizierte Identitäten aus einem Amazon Cognito Cognito-Benutzerpool zugreifen kann.

Die folgende `config.txt` Datei enthält die Details des Amazon Cognito Cognito-Benutzerpools, der vom Parameter --configuration im `create-identity-source` Befehl verwendet werden kann.

```
{
    "cognitoUserPoolConfiguration": {
        "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_1a2b3c4d5",
```

```
        "clientIds":["a1b2c3d4e5f6g7h8i9j0kalbmc"],
        "groupConfiguration": {
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

Befehl:

```
$ aws verifiedpermissions create-identity-source \
    --configuration file://config.txt \
    --principal-entity-type "User" \
    --policy-store-id 123456789012
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

Wenn Ihr Richtlinienspeicher über ein Schema verfügt, müssen Sie, bevor Sie in Ihren Cedar-Richtlinien auf Attribute verweisen können, die Sie aus Identitäts- oder Zugriffstoken extrahieren, Ihr Schema aktualisieren, damit Cedar weiß, welche Art von Prinzipal Ihre Identitätsquelle erstellt. Diese Ergänzung zum Schema muss die Attribute enthalten, auf die Sie in Ihren Cedar-Richtlinien verweisen möchten. Weitere Informationen zur Zuordnung von Amazon Cognito-Tokenattributen zu Cedar-Hauptattributen finden Sie unterZuordnen von Identitätsanbieter-Tokens zum Schema.

Wenn Sie einen API-verknüpften Richtlinienspeicher erstellen oder beim Erstellen von Richtlinienspeichern die Option Setup with API Gateway und Identity Provider verwenden, fragt Verified Permissions Ihren Benutzerpool nach Benutzerattributen ab und erstellt ein Schema, in dem Ihr Prinzipaltyp mit Benutzerpool-Attributen gefüllt wird.

Weitere Informationen zur Verwendung von Zugriffs- und Identitätstoken von Amazon Cognito für authentifizierte Benutzer in Verified Permissions finden Sie unter Authorization with Amazon Verified Permissions im Amazon Cognito Developer Guide.

OIDC-Identitätsquelle

AWS Management Console

So erstellen Sie eine OpenID Connect (OIDC) -Identitätsquelle

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Identitätsquellen aus.
3. Wählen Sie Identitätsquelle erstellen aus.
4. Wählen Sie Externer OIDC-Anbieter.
5. Geben Sie im Feld Aussteller-URL die URL Ihres OIDC-Ausstellers ein. Dies ist der Dienstendpunkt, der beispielsweise den Autorisierungsserver, Signaturschlüssel und andere Informationen über Ihren Anbieter bereitstellt. `https://auth.example.com` Ihre Aussteller-URL muss ein OIDC-Discovery-Dokument unter hosten. `/.well-known/openid-configuration`
6. Wählen Sie unter Tokentyp den Typ des OIDC JWT aus, den Ihre Anwendung zur Autorisierung einreichen soll. Weitere Informationen finden Sie unter Zuordnen von Identitätsanbieter-Tokens zum Schema.
7. Wählen Sie unter Tokenansprüche Schemaentitäten zuordnen eine Benutzerentität und Benutzeranspruch für die Identitätsquelle aus. Die Benutzerentität ist eine Entität in Ihrem Richtlinienspeicher, auf die Sie auf Benutzer Ihres OIDC-Anbieters verweisen möchten. Bei dem Benutzeranspruch handelt es sich in der Regel `sub` um einen Anspruch aus Ihrer ID oder Ihrem Zugriffstoken, das die eindeutige Kennung für die zu bewertende Entität enthält. Identitäten des verbundenen OIDC-IdP werden dem ausgewählten Prinzipaltyp zugeordnet.
8. (Optional) Wählen Sie unter Tokenansprüche Schemaentitäten zuordnen eine Gruppenentität und einen Gruppenanspruch für die Identitätsquelle aus. Die Gruppenentität ist der Benutzerentität übergeordnet. Gruppenansprüche werden dieser Entität zugeordnet. Bei dem Gruppenanspruch handelt es sich in der Regel `groups` um einen Anspruch aus Ihrer ID oder Ihrem Zugriffstoken, der eine Zeichenfolge, JSON oder eine durch Leerzeichen getrennte Zeichenfolge mit Benutzergruppennamen für die auszuwertende Entität enthält. Identitäten des verbundenen OIDC-IdP werden dem ausgewählten Prinzipaltyp zugeordnet.
9. Geben Sie im Feld Validierung — optional den Kunden IDs oder die Zielgruppe ein URLs , die Ihr Richtlinienspeicher gegebenenfalls in Autorisierungsanfragen akzeptieren soll.
10. Wählen Sie „Identitätsquelle erstellen“.

11. Aktualisieren Sie Ihr Schema, damit Cedar weiß, welchen Prinzipaltyp Ihre Identitätsquelle erstellt. Dieser Zusatz zum Schema muss die Attribute enthalten, auf die Sie in Ihren Cedar-Richtlinien verweisen möchten. Weitere Informationen zur Zuordnung von Amazon Cognito-Tokenattributen zu Cedar-Hauptattributen finden Sie unterZuordnen von Identitätsanbieter-Tokens zum Schema.

    Wenn Sie einen API-verknüpften Richtlinienspeicher erstellen, fragt Verified Permissions Ihren Benutzerpool nach Benutzerattributen ab und erstellt ein Schema, in dem Ihr Prinzipaltyp mit Benutzerpool-Attributen aufgefüllt wird.

AWS CLI

Um eine OIDC-Identitätsquelle zu erstellen

Sie können eine Identitätsquelle erstellen, indem Sie den CreateIdentitySourceVorgang verwenden. Im folgenden Beispiel wird eine Identitätsquelle erstellt, die auf authentifizierte Identitäten aus einem Amazon Cognito Cognito-Benutzerpool zugreifen kann.

Die folgende `config.txt` Datei enthält die Details eines OIDC-IdP zur Verwendung durch den `--configuration` Parameter des Befehls. `create-identity-source` In diesem Beispiel wird eine OIDC-Identitätsquelle für ID-Token erstellt.

```
{
    "openIdConnectConfiguration": {
        "issuer": "https://auth.example.com",
        "tokenSelection": {
                "identityTokenOnly": {
                        "clientIds":["1example23456789"],
                        "principalIdClaim": "sub"
                },
        },
        "entityIdPrefix": "MyOIDCProvider",
        "groupConfiguration": {
              "groupClaim": "groups",
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

Die folgende `config.txt` Datei enthält die Details eines OIDC-IdP zur Verwendung durch den `--configuration` Parameter des Befehls. `create-identity-source` In diesem Beispiel wird eine OIDC-Identitätsquelle für Zugriffstoken erstellt.

```
{
    "openIdConnectConfiguration": {
        "issuer": "https://auth.example.com",
        "tokenSelection": {
                "accessTokenOnly": {
                        "audiences":["https://auth.example.com"],
                        "principalIdClaim": "sub"
                },
        },
        "entityIdPrefix": "MyOIDCProvider",
        "groupConfiguration": {
              "groupClaim": "groups",
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

Befehl:

```
$ aws verifiedpermissions create-identity-source \
    --configuration file://config.txt \
    --principal-entity-type "User" \
    --policy-store-id 123456789012
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

Bevor Sie in Ihren Cedar-Richtlinien auf Attribute verweisen können, die Sie aus Identitäts- oder Zugriffstoken extrahieren, müssen Sie Ihr Schema aktualisieren, damit Cedar weiß, welche Art von Prinzipal Ihre Identitätsquelle erstellt. Diese Ergänzung zum Schema muss die Attribute enthalten, auf die Sie in Ihren Cedar-Richtlinien verweisen möchten. Weitere Informationen zur Zuordnung von Amazon Cognito-Tokenattributen zu Cedar-Hauptattributen finden Sie unterZuordnen von Identitätsanbieter-Tokens zum Schema.

Wenn Sie einen API-verknüpften Richtlinienspeicher erstellen, fragt Verified Permissions Ihren Benutzerpool nach Benutzerattributen ab und erstellt ein Schema, in dem Ihr Prinzipaltyp mit Benutzerpool-Attributen aufgefüllt wird.

Identitätsquellen für Amazon Verified Permissions bearbeiten

Sie können einige Parameter Ihrer Identitätsquelle bearbeiten, nachdem Sie sie erstellt haben. Sie können den Typ der Identitätsquelle nicht ändern. Sie müssen die Identitätsquelle löschen und eine neue erstellen, um von Amazon Cognito zu OIDC oder OIDC zu Amazon Cognito zu wechseln. Wenn Ihr Richtlinienspeicherschema Ihren Identitätsquellenattributen entspricht, beachten Sie, dass Sie Ihr Schema separat aktualisieren müssen, um die Änderungen widerzuspiegeln, die Sie an Ihrer Identitätsquelle vornehmen.

Themen

- Identitätsquelle für Amazon Cognito Cognito-Benutzerpools
- OpenID Connect (OIDC) -Identitätsquelle

Identitätsquelle für Amazon Cognito Cognito-Benutzerpools

AWS Management Console

So aktualisieren Sie die Identitätsquelle eines Amazon Cognito Cognito-Benutzerpools

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Identitätsquellen aus.
3. Wählen Sie die ID der Identitätsquelle aus, die Sie bearbeiten möchten.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Wählen Sie in den Cognito-Benutzerpooldetails die Benutzerpool-ID für Ihre Identitätsquelle aus AWS-Region und geben Sie sie ein.
6. In den Prinzipaldetails können Sie den Prinzipaltyp für die Identitätsquelle aktualisieren. Identitäten aus den verbundenen Amazon Cognito Cognito-Benutzerpools werden dem ausgewählten Prinzipaltyp zugeordnet.
7. Wählen Sie in der Gruppenkonfiguration die Option Cognito-Gruppen verwenden aus, wenn Sie den `cognito:groups` Benutzerpoolanspruch zuordnen möchten. Wählen Sie einen Entitätstyp, der dem Prinzipaltyp übergeordnet ist.

8. Wählen Sie unter Validierung der Client-Anwendung aus, ob die Client-Anwendung validiert werden soll IDs.

   - Um die Client-Anwendung zu validieren IDs, wählen Sie Nur Tokens mit passender Client-Anwendung akzeptieren IDs. Wählen Sie Neue Client-Anwendungs-ID hinzufügen für jede zu validierende Client-Anwendungs-ID aus. Um eine hinzugefügte Client-Anwendungs-ID zu entfernen, klicken Sie neben der Client-Anwendungs-ID auf Entfernen.
   - Wählen Sie Client-Anwendung nicht validieren IDs, wenn Sie die Client-Anwendung nicht validieren möchten IDs.
9. Wählen Sie Änderungen speichern.
10. Wenn Sie den Prinzipaltyp für die Identitätsquelle geändert haben, müssen Sie Ihr Schema aktualisieren, damit es den aktualisierten Prinzipaltyp korrekt wiedergibt.

Sie können eine Identitätsquelle löschen, indem Sie das Optionsfeld neben einer Identitätsquelle auswählen und dann Identitätsquelle löschen auswählen. Geben Sie `delete` etwas in das Textfeld ein und wählen Sie dann Identitätsquelle löschen aus, um das Löschen der Identitätsquelle zu bestätigen.

AWS CLI

So aktualisieren Sie die Identitätsquelle eines Amazon Cognito Cognito-Benutzerpools

Sie können eine Identitätsquelle aktualisieren, indem Sie den UpdateIdentitySourceVorgang verwenden. Im folgenden Beispiel wird die angegebene Identitätsquelle aktualisiert, sodass sie einen anderen Amazon Cognito Cognito-Benutzerpool verwendet.

Die folgende `config.txt` Datei enthält die Details des Amazon Cognito Cognito-Benutzerpools zur Verwendung durch den Parameter --configuration im `create-identity-source` Befehl.

```
{
    "cognitoUserPoolConfiguration": {
        "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_1a2b3c4d5",
        "clientIds":["a1b2c3d4e5f6g7h8i9j0kalbmc"],
        "groupConfiguration": {
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

Befehl:

```
$ aws verifiedpermissions update-identity-source \
    --update-configuration file://config.txt \
    --policy-store-id 123456789012
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

Wenn Sie den Prinzipaltyp für die Identitätsquelle ändern, müssen Sie Ihr Schema aktualisieren, damit es den aktualisierten Prinzipaltyp korrekt wiedergibt.

OpenID Connect (OIDC) -Identitätsquelle

AWS Management Console

Um eine OIDC-Identitätsquelle zu aktualisieren

1. Öffnen Sie die Konsole Verified Permissions. Wählen Sie Ihren Richtlinienspeicher aus.
2. Wählen Sie im Navigationsbereich auf der linken Seite Identitätsquellen aus.
3. Wählen Sie die ID der Identitätsquelle aus, die Sie bearbeiten möchten.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Ändern Sie in den OIDC-Anbieterdetails die Aussteller-URL nach Bedarf.
6. Ändern Sie unter Tokenansprüche den Schemaattributen zuordnen die Verknüpfungen zwischen Benutzer- und Gruppenansprüchen sowie den Entitätstypen des Richtlinienspeichers nach Bedarf. Nachdem Sie die Entitätstypen geändert haben, müssen Sie Ihre Richtlinien und Schemaattribute aktualisieren, damit sie für die neuen Entitätstypen gelten.
7. Fügen Sie in der Zielgruppenvalidierung Zielgruppenwerte hinzu oder entfernen Sie sie, die Sie erzwingen möchten.
8. Wählen Sie Änderungen speichern.

Sie können eine Identitätsquelle löschen, indem Sie das Optionsfeld neben einer Identitätsquelle auswählen und dann Identitätsquelle löschen auswählen. Geben Sie `delete` etwas in

das Textfeld ein und wählen Sie dann Identitätsquelle löschen aus, um das Löschen der Identitätsquelle zu bestätigen.

AWS CLI

Um eine OIDC-Identitätsquelle zu aktualisieren

Sie können eine Identitätsquelle aktualisieren, indem Sie den UpdateIdentitySourceVorgang verwenden. Im folgenden Beispiel wird die angegebene Identitätsquelle aktualisiert, sodass sie einen anderen OIDC-Anbieter verwendet.

Die folgende `config.txt` Datei enthält die Details des Amazon Cognito Cognito-Benutzerpools zur Verwendung durch den Parameter --configuration im `create-identity-source` Befehl.

```
{
    "openIdConnectConfiguration": {
        "issuer": "https://auth2.example.com",
        "tokenSelection": {
                "identityTokenOnly": {
                        "clientIds":["2example10111213"],
                        "principalIdClaim": "sub"
                },
        },
        "entityIdPrefix": "MyOIDCProvider",
        "groupConfiguration": {
              "groupClaim": "groups",
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

Befehl:

```
$ aws verifiedpermissions update-identity-source \
    --update-configuration file://config.txt \
    --policy-store-id 123456789012
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

Wenn Sie den Prinzipaltyp für die Identitätsquelle ändern, müssen Sie Ihr Schema aktualisieren, damit es den aktualisierten Prinzipaltyp korrekt wiedergibt.

Zuordnen von Identitätsanbieter-Tokens zum Schema

Möglicherweise möchten Sie einem Richtlinienspeicher eine Identitätsquelle hinzufügen und Anbieteransprüche oder Token Ihrem Richtlinienspeicherschema zuordnen. Sie können diesen Vorgang automatisieren, indem Sie den geführten Einrichtungsvorgang verwenden, um Ihren Richtlinienspeicher mit einer Identitätsquelle zu erstellen, oder Ihr Schema manuell aktualisieren, nachdem der Richtlinienspeicher erstellt wurde. Sobald Sie die Token dem Schema zugeordnet haben, können Sie Richtlinien erstellen, die auf sie verweisen.

Dieser Abschnitt des Benutzerhandbuchs enthält die folgenden Informationen:

- Wann können Sie Attribute automatisch in ein Policy-Store-Schema eintragen
- So verwenden Sie Amazon Cognito- und OIDC-Token-Ansprüche in Ihren Richtlinien für verifizierte Berechtigungen
- Wie erstelle ich manuell ein Schema für eine Identitätsquelle

API-verknüpfte Richtlinienspeicher und Richtlinienspeicher mit einer Identitätsquelle, die über die geführte Installation erstellt wurden, erfordern keine manuelle Zuordnung von Identitätstokenattributen (ID) zum Schema. Sie können den Attributen in Ihrem Benutzerpool verifizierte Berechtigungen zuordnen und ein Schema erstellen, das mit Benutzerattributen gefüllt ist. Bei der Autorisierung von ID-Tokens ordnet Verified Permissions Ansprüche Attributen einer Prinzipalentität zu. Unter den folgenden Bedingungen müssen Sie Amazon Cognito Cognito-Token möglicherweise manuell Ihrem Schema zuordnen:

- Sie haben einen leeren Richtlinienspeicher oder Richtlinienspeicher anhand eines Beispiels erstellt.
- Sie möchten Ihre Verwendung von Zugriffstoken über die rollenbasierte Zugriffskontrolle (RBAC) hinaus erweitern.
- Sie erstellen Richtlinienspeicher mit der REST-API für verifizierte Berechtigungen, einem AWS SDK oder dem. AWS CDK

Um Amazon Cognito oder einen OIDC-Identitätsanbieter (IdP) als Identitätsquelle in Ihrem Richtlinienspeicher für verifizierte Berechtigungen zu verwenden, müssen Sie Anbieterattribute in Ihrem Schema haben. Das Schema ist fest und muss den Entitäten entsprechen, die Anbieter-

Token in IsAuthorizedWithTokenAPI-Anfragen erstellen. BatchIsAuthorizedWithToken Wenn Sie Ihren Richtlinienspeicher so erstellt haben, dass Ihr Schema automatisch anhand der Anbieterinformationen in einem ID-Token aufgefüllt wird, sind Sie bereit, Richtlinien zu schreiben. Wenn Sie einen Richtlinienspeicher ohne Schema für Ihre Identitätsquelle erstellen, müssen Sie dem Schema Anbieterattribute hinzufügen, die den mithilfe von API-Anfragen erstellten Entitäten entsprechen. Anschließend können Sie Richtlinien mithilfe von Attributen aus dem Anbietertoken schreiben.

Weitere Informationen zur Verwendung von Amazon Cognito ID und Zugriffstoken für authentifizierte Benutzer in Verified Permissions finden Sie unter Authorization with Amazon Verified Permissions im Amazon Cognito Developer Guide.

Themen

- Zuordnen von ID-Token zum Schema
- Zugriffstoken zuordnen
- Alternative Schreibweise für durch Doppelpunkte getrennte Amazon Cognito-Ansprüche
- Wissenswertes über Schema-Mapping

Zuordnen von ID-Token zum Schema

Verified Permissions verarbeitet ID-Token-Ansprüche als Attribute des Benutzers: seine Namen und Titel, seine Gruppenzugehörigkeit, seine Kontaktinformationen. ID-Token sind in einem ABAC-Autorisierungsmodell (attribute-based access control) am nützlichsten. Wenn Sie möchten, dass Verified Permissions den Zugriff auf Ressourcen basierend darauf analysiert, wer die Anfrage stellt, wählen Sie ID-Token als Identitätsquelle.

Amazon Cognito Cognito-ID-Token

Amazon Cognito ID-Token funktionieren mit den meisten OIDC-Rely-Party-Bibliotheken. Sie erweitern die Funktionen von OIDC um zusätzliche Ansprüche. Ihre Anwendung kann den Benutzer mit API-Authentifizierungsoperationen für Amazon Cognito Cognito-Benutzerpools oder mit der gehosteten Benutzerpool-Benutzeroberfläche authentifizieren. Weitere Informationen finden Sie unter Using the API and Endpoints im Amazon Cognito Developer Guide.

Nützliche Angaben in Amazon Cognito Cognito-ID-Tokens

cognito:username und *preferred_username*

Varianten des Benutzernamens.

sub

Die eindeutige Benutzerkennung (UUID) des Benutzers

Ansprüche mit einem Präfix *custom:*

Ein Präfix für benutzerdefinierte Benutzerpool-Attribute wie`custom:employmentStoreCode`.

Standardansprüche

Standardansprüche von OIDC wie `email` und. `phone_number` Weitere Informationen finden Sie unter Standardansprüche in OpenID Connect Core 1.0, in denen Errata Set 2 enthalten ist.

cognito:groups

Die Gruppenmitgliedschaften eines Benutzers. In einem Autorisierungsmodell, das auf der rollenbasierten Zugriffskontrolle (RBAC) basiert, stellt dieser Anspruch die Rollen dar, die Sie in Ihren Richtlinien bewerten können.

Vorübergehende Ansprüche

Ansprüche, die nicht Eigentum des Benutzers sind, aber zur Laufzeit durch einen Lambda-Trigger vor der Token-Generierung aus dem Benutzerpool hinzugefügt werden. Vorübergehende Ansprüche ähneln Standardansprüchen, liegen aber außerhalb des Standards, z. B. `tenant` oder. `department`

In Richtlinien, die auf Amazon Cognito-Attribute verweisen, die über ein `:` Trennzeichen verfügen, verweisen Sie auf die Attribute im Format`principal["cognito:username"]`. Der Rollenanspruch `cognito:groups` stellt eine Ausnahme von dieser Regel dar. Verified Permissions ordnet den Inhalt dieses Anspruchs den übergeordneten Entitäten der Benutzerentität zu.

Weitere Informationen zur Struktur von ID-Token aus Amazon Cognito-Benutzerpools finden Sie unter Verwenden des ID-Tokens im Amazon Cognito Developer Guide.

Das folgende Beispiel für ein ID-Token hat jeden der vier Attributtypen. Es umfasst den Amazon Cognito-spezifischen Antrag`cognito:username`, den benutzerdefinierten Antrag, den

Standardantrag `custom:employmentStoreCode` und den `email` vorübergehenden Anspruch. `tenant`

```
{
    "sub": "91eb4550-XXX",
    "cognito:groups": [
        "Store-Owner-Role",
        "Customer"
    ],
    "email_verified": true,
    "clearance": "confidential",
    "iss": "https://cognito-idp.us-east-2.amazonaws.com/us-east-2_EXAMPLE",
    "cognito:username": "alice",
    "custom:employmentStoreCode": "petstore-dallas",
    "origin_jti": "5b9f50a3-05da-454a-8b99-b79c2349de77",
    "aud": "1example23456789",
    "event_id": "0ed5ad5c-7182-4ecf-XXX",
    "token_use": "id",
    "auth_time": 1687885407,
    "department": "engineering",
    "exp": 1687889006,
    "iat": 1687885407,
    "tenant": "x11app-tenant-1",
    "jti": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "email": "alice@example.com"
}
```

Wenn Sie mit Ihrem Amazon Cognito Cognito-Benutzerpool eine Identitätsquelle erstellen, geben Sie den Typ der Prinzipalentität an, mit `IsAuthorizedWithToken` der Verified Permissions in Autorisierungsanfragen generiert. Ihre Richtlinien können dann im Rahmen der Bewertung dieser Anfrage die Attribute dieses Prinzipals testen. Ihr Schema definiert den Prinzipaltyp und die Attribute für eine Identitätsquelle, und dann können Sie in Ihren Cedar-Richtlinien darauf verweisen.

Sie geben auch den Typ der Gruppenentität an, den Sie aus dem Anspruch der ID-Token-Gruppen ableiten möchten. In Autorisierungsanfragen ordnet Verified Permissions jedes Mitglied des Gruppenanspruchs diesem Gruppen-Entitätstyp zu. In Richtlinien können Sie auf diese Gruppenentität als Principal verweisen.

Das folgende Beispiel zeigt, wie Sie die Attribute aus dem Beispiel-Identitätstoken in Ihrem Verified Permissions-Schema wiedergeben können. Weitere Informationen zur Bearbeitung Ihres Schemas finden Sie unter[Policy-Store-Schemas bearbeiten](). Wenn Ihre Identitätsquellenkonfiguration den

Prinzipaltyp angibt`User`, können Sie etwas Ähnliches wie das folgende Beispiel hinzufügen, um diese Attribute für Cedar verfügbar zu machen.

```
"User": {
   "shape": {
      "type": "Record",
      "attributes": {
         "cognito:username": {
            "type": "String",
            "required": false
         },
         "custom:employmentStoreCode": {
            "type": "String",
            "required": false
         },
         "email": {
            "type": "String"
         },
         "tenant": {
            "type": "String",
            "required": true
         }
      }
   }
}
```

Ein Beispiel für eine Richtlinie, die anhand dieses Schemas validiert wird, finden Sie unterSpiegelt Amazon Cognito Cognito-ID-Token-Attribute wider.

OIDC-ID-Token

Die Arbeit mit ID-Token von einem OIDC-Anbieter entspricht weitgehend der Arbeit mit Amazon Cognito Cognito-ID-Token. Der Unterschied liegt in den Behauptungen. Ihr IdP kann Standard-OIDC-Attribute oder ein benutzerdefiniertes Schema aufweisen. Wenn Sie in der Konsole „Verified Permissions“ einen neuen Richtlinienspeicher erstellen, können Sie eine OIDC-Identitätsquelle mit einem Beispiel-ID-Token hinzufügen oder Tokenansprüche manuell Benutzerattributen zuordnen. Da Verified Permissions das Attributschema Ihres IdP nicht kennt, müssen Sie diese Informationen angeben.

Weitere Informationen finden Sie unter Richtlinienspeicher für verifizierte Berechtigungen erstellen.

Im Folgenden finden Sie ein Beispielschema für einen Richtlinienspeicher mit einer OIDC-Identitätsquelle.

```
"User": {
   "shape": {
      "type": "Record",
      "attributes": {
         "email": {
            "type": "String"
         },
         "email_verified": {
            "type": "Boolean"
         },
         "name": {
            "type": "String",
            "required": true
         },
         "phone_number": {
            "type": "String"
         },
         "phone_number_verified": {
            "type": "Boolean"
         }
      }
   }
}
```

Ein Beispiel für eine Richtlinie, die anhand dieses Schemas validiert wird, finden Sie unter. Spiegelt die OIDC-ID-Token-Attribute wider

Zugriffstoken zuordnen

Verified Permissions verarbeitet Ansprüche auf Zugriffstoken, die nicht von Gruppen als Attribute der Aktion oder als Kontext-Attribute beansprucht werden. Neben der Gruppenmitgliedschaft können die Zugriffstoken Ihres IdP Informationen über den API-Zugriff enthalten. Zugriffstoken sind in Autorisierungsmodellen nützlich, die eine rollenbasierte Zugriffskontrolle (RBAC) verwenden. Autorisierungsmodelle, die auf anderen Zugriffstoken-Ansprüchen als der Gruppenmitgliedschaft basieren, erfordern zusätzlichen Aufwand bei der Schemakonfiguration.

Zuordnen von Amazon Cognito Cognito-Zugriffstoken

Amazon Cognito Cognito-Zugriffstoken haben Ansprüche, die für die Autorisierung verwendet werden können:

Nützliche Angaben in Amazon Cognito Cognito-Zugriffstoken

`client_id`

Die ID der Client-Anwendung einer vertrauenden OIDC-Partei. Anhand der Client-ID kann Verified Permissions überprüfen, ob die Autorisierungsanfrage von einem autorisierten Client für den Richtlinienspeicher stammt. Bei der machine-to-machine (M2M-) Autorisierung autorisiert das anfordernde System eine Anfrage mit einem geheimen Client-Schlüssel und stellt die Client-ID und den Geltungsbereich als Autorisierungsnachweis zur Verfügung.

`scope`

Die OAuth 2.0-Bereiche, die die Zugriffsberechtigungen des Inhabers des Tokens darstellen.

`cognito:groups`

Die Gruppenmitgliedschaften eines Benutzers. In einem Autorisierungsmodell, das auf der rollenbasierten Zugriffskontrolle (RBAC) basiert, stellt dieser Anspruch die Rollen dar, die Sie in Ihren Richtlinien bewerten können.

Vorübergehende Ansprüche

Ansprüche, bei denen es sich nicht um eine Zugriffsberechtigung handelt, die aber zur Laufzeit durch einen Lambda-Trigger vor der Token-Generierung im Benutzerpool hinzugefügt werden. Vorübergehende Ansprüche ähneln Standardansprüchen, liegen aber außerhalb des Standards, z. B. `tenant` oder. `department` Die Anpassung von Zugriffstoken erhöht Ihre AWS Rechnung um zusätzliche Kosten.

Weitere Informationen zur Struktur von Zugriffstoken aus Amazon Cognito-Benutzerpools finden Sie unter Verwenden des Zugriffstokens im Amazon Cognito Developer Guide.

Ein Amazon Cognito Cognito-Zugriffstoken wird einem Kontextobjekt zugeordnet, wenn es an Verified Permissions übergeben wird. Auf Attribute des Zugriffstokens kann mit verwiesen werden. `context.token.`*`attribute_name`* Das folgende Beispiel für ein Zugriffstoken umfasst `client_id` sowohl die `scope` Ansprüche als auch.

```
{
```

```
    "sub": "91eb4550-9091-708c-a7a6-9758ef8b6b1e",
    "cognito:groups": [
        "Store-Owner-Role",
        "Customer"
    ],
    "iss": "https://cognito-idp.us-east-2.amazonaws.com/us-east-2_EXAMPLE",
    "client_id": "1example23456789",
    "origin_jti": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "event_id": "bda909cb-3e29-4bb8-83e3-ce6808f49011",
    "token_use": "access",
    "scope": "MyAPI/mydata.write",
    "auth_time": 1688092966,
    "exp": 1688096566,
    "iat": 1688092966,
    "jti": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN2222222",
    "username": "alice"
}
```

Das folgende Beispiel zeigt, wie Sie die Attribute aus dem Beispielzugriffstoken in Ihrem Verified Permissions-Schema wiedergeben können. Weitere Informationen zur Bearbeitung Ihres Schemas finden Sie unterPolicy-Store-Schemas bearbeiten.

```
{
   "MyApplication": {
      "actions": {
         "Read": {
            "appliesTo": {
               "context": {
                  "type": "ReusedContext"
               },
               "resourceTypes": [
                  "Application"
               ],
               "principalTypes": [
                  "User"
               ]
            }
         }
      },
      ...
      ...
      "commonTypes": {
         "ReusedContext": {
```

```
            "attributes": {
               "token": {
                  "type": "Record",
                  "attributes": {
                     "scope": {
                        "type": "Set",
                        "element": {
                           "type": "String"
                        }
                     },
                     "client_id": {
                        "type": "String"
                     }
                  }
               }
            },
            "type": "Record"
         }
      }
   }
}
```

Ein Beispiel für eine Richtlinie, die anhand dieses Schemas validiert wird, finden Sie unterSpiegelt die Attribute des Amazon Cognito Cognito-Zugriffstokens wider.

Zuordnen von OIDC-Zugriffstoken

Die meisten Zugriffstoken von externen OIDC-Anbietern stimmen eng mit den Zugriffstoken von Amazon Cognito überein. Ein OIDC-Zugriffstoken wird einem Kontextobjekt zugeordnet, wenn es an Verified Permissions übergeben wird. Auf Attribute des Zugriffstokens kann mit verwiesen werden. `context.token.`*`attribute_name`* Das folgende Beispiel für ein OIDC-Zugriffstoken enthält Beispiele für Basisansprüche.

```
{
    "sub": "91eb4550-9091-708c-a7a6-9758ef8b6b1e",
    "groups": [
        "Store-Owner-Role",
        "Customer"
    ],
    "iss": "https://auth.example.com",
    "client_id": "1example23456789",
    "aud": "https://myapplication.example.com"
```

```
    "scope": "MyAPI-Read",
    "exp": 1688096566,
    "iat": 1688092966,
    "jti": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN2222222",
    "username": "alice"
}
```

Das folgende Beispiel zeigt, wie Sie die Attribute aus dem Beispiel-Zugriffstoken in Ihrem Verified Permissions-Schema wiedergeben können. Weitere Informationen zur Bearbeitung Ihres Schemas finden Sie unterPolicy-Store-Schemas bearbeiten.

```
{
   "MyApplication": {
      "actions": {
         "Read": {
            "appliesTo": {
               "context": {
                  "type": "ReusedContext"
               },
               "resourceTypes": [
                  "Application"
               ],
               "principalTypes": [
                  "User"
               ]
            }
         }
      },
      ...
      ...
      "commonTypes": {
         "ReusedContext": {
            "attributes": {
               "token": {
                  "type": "Record",
                  "attributes": {
                     "scope": {
                        "type": "Set",
                        "element": {
                           "type": "String"
                        }
                     },
                     "client_id": {
```

```
                        "type": "String"
                    }
                }
            }
        },
        "type": "Record"
    }
  }
}
```

Ein Beispiel für eine Richtlinie, die anhand dieses Schemas validiert wird, finden Sie unterSpiegelt die Attribute von OIDC-Zugriffstoken wider.

Alternative Schreibweise für durch Doppelpunkte getrennte Amazon Cognito-Ansprüche

Zu dem Zeitpunkt, als Verified Permissions gestartet wurde, beanspruchte das empfohlene Schema für Amazon Cognito Cognito-Token „Gefällt mir“ `cognito:groups` und diese durch Doppelpunkte getrennten Zeichenketten wurden `custom:store` konvertiert, um das `.` Zeichen als Hierarchie-Trennzeichen zu verwenden. Dieses Format wird als Punktnotation bezeichnet. Beispielsweise `cognito:groups` wurde `principal.cognito.groups` in Ihren Richtlinien ein Verweis auf aufgenommen. Sie können dieses Format zwar weiterhin verwenden, wir empfehlen Ihnen jedoch, Ihr Schema und Ihre Richtlinien in Klammern zu erstellen. In diesem Format `cognito:groups` wird ein Verweis auf `principal["cognito:groups"]` in Ihren Richtlinien enthalten. Automatisch generierte Schemas für Benutzerpool-ID-Token aus der Verified Permissions-Konsole verwenden die Schreibweise in Klammern.

Sie können weiterhin die Punktnotation in manuell erstellten Schemas und Richtlinien für Amazon Cognito Cognito-Identitätsquellen verwenden. Sie können die Punktnotation mit `:` oder anderen nicht alphanumerischen Zeichen in Schemas oder Richtlinien für keinen anderen Typ von OIDC-IdP verwenden.

Ein Schema für die Punktnotation verschachtelt jede Instanz eines `:` Zeichens als untergeordnetes Element der `cognito` oder `custom` ersten Phrase, wie im folgenden Beispiel gezeigt:

```
"CognitoUser": {
   "shape": {
      "type": "Record",
```

```
      "attributes": {
         "cognito": {
            "type": "Record",
            "required": true,
            "attributes": {
               "username": {
                  "type": "String",
                  "required": true
               }
            }
         },
         "custom": {
            "type": "Record",
            "required": true,
            "attributes": {
               "employmentStoreCode": {
                  "type": "String",
                  "required": true
               }
            }
         },
         "email": {
            "type": "String"
         },
         "tenant": {
            "type": "String",
            "required": true
         }
      }
   }
}
```

Ein Beispiel für eine Richtlinie, die anhand dieses Schemas validiert und die Punktnotation verwendet, finden Sie unterVerwendet Punktnotation, um auf Attribute zu verweisen.

Wissenswertes über Schema-Mapping

Die Attributzuweisung unterscheidet sich zwischen Tokentypen

Bei der Autorisierung von Zugriffstoken ordnet Verified Permissions Ansprüche dem Kontext zu. Bei der Autorisierung von ID-Tokens ordnet Verified Permissions Ansprüche den Hauptattributen zu. Bei Richtlinienspeichern, die Sie in der Verified Permissions-Konsole erstellen, haben Sie nur leere Richtlinienspeicher und Beispiel-Richtlinienspeicher, sodass Sie keine Identitätsquelle

haben und Sie Ihr Schema mit Benutzerpool-Attributen für die ID-Token-Autorisierung auffüllen müssen. Die Autorisierung mit Zugriffstoken basiert auf der rollenbasierten Zugriffskontrolle (RBAC) mit Gruppenmitgliedschaftsansprüchen und ordnet andere Ansprüche nicht automatisch dem Richtlinienspeicherschema zu.

Identitätsquellenattribute sind nicht erforderlich

Wenn Sie in der Konsole „Verified Permissions“ eine Identitätsquelle erstellen, werden keine Attribute als erforderlich markiert. Dadurch wird verhindert, dass fehlende Ansprüche zu Validierungsfehlern bei Autorisierungsanfragen führen. Sie können Attribute nach Bedarf auf erforderlich setzen, sie müssen jedoch in allen Autorisierungsanfragen vorhanden sein.

RBAC benötigt keine Attribute im Schema

Schemas für Identitätsquellen hängen von den Entitätszuordnungen ab, die Sie beim Hinzufügen Ihrer Identitätsquelle vornehmen. Eine Identitätsquelle ordnet einen Anspruch einem Benutzerentiätstyp und einen Anspruch einem Gruppen-Entitätstyp zu. Diese Entitätszuordnungen sind der Kern einer Identitätsquellenkonfiguration. Mit diesen Mindestinformationen können Sie Richtlinien schreiben, die Autorisierungsaktionen für bestimmte Benutzer und bestimmte Gruppen, denen Benutzer möglicherweise angehören, in einem Modell der rollenbasierten Zugriffskontrolle (RBAC) ausführen. Durch das Hinzufügen von Token-Ansprüchen zum Schema wird der Autorisierungsbereich Ihres Richtlinienspeichers erweitert. Benutzerattribute aus ID-Tokens enthalten Informationen über Benutzer, die zur ABAC-Autorisierung (attribute-Based Access Control) beitragen können. Kontextattribute von Zugriffstoken enthalten Informationen wie OAuth 2.0-Bereiche, die zusätzliche Informationen zur Zugriffskontrolle von Ihrem Anbieter bereitstellen können, aber zusätzliche Schemaänderungen erfordern.

Die Optionen Mit API Gateway und einem Identitätsanbieter einrichten und Geführte Einrichtung in der Konsole Verified Permissions weisen dem Schema ID-Token-Ansprüche zu. Dies ist bei Ansprüchen auf Zugriffstoken nicht der Fall. Um Ihrem Schema Ansprüche auf Zugriffstoken hinzuzufügen, die nicht zu Gruppen gehören, müssen Sie Ihr Schema im JSON-Modus bearbeiten und CommonTypes-Attribute hinzufügen. Weitere Informationen finden Sie unter Zugriffstoken zuordnen.

OIDC-Gruppen behaupten, dass mehrere Formate unterstützt werden

Wenn Sie einen OIDC-Anbieter hinzufügen, können Sie den Namen des Gruppenanspruchs in ID- oder Zugriffstoken auswählen, den Sie der Gruppenmitgliedschaft eines Benutzers in Ihrem Richtlinienspeicher zuordnen möchten. Verifizierte Berechtigungen erkennen Gruppenansprüche in den folgenden Formaten an:

1. Zeichenfolge ohne Leerzeichen: `"groups": "MyGroup"`
2. Durch Leerzeichen getrennte Liste:. `"groups": "MyGroup1 MyGroup2 MyGroup3"` Jede Zeichenfolge ist eine Gruppe.
3. JSON-Liste (durch Kommas getrennt): `"groups": ["MyGroup1", "MyGroup2", "MyGroup3"]`

> **Note**
>
> Verified Permissions interpretiert jede Zeichenfolge in einem durch Leerzeichen getrennten Gruppenanspruch als separate Gruppe. Um einen Gruppennamen mit einem Leerzeichen als einzelne Gruppe zu interpretieren, ersetzen oder entfernen Sie das Leerzeichen im Anspruch. Formatieren Sie beispielsweise eine Gruppe mit dem Namen `My GroupMyGroup`.

Wählen Sie einen Tokentyp

Wie Ihr Richtlinienspeicher mit Ihrer Identitätsquelle zusammenarbeitet, hängt von einer wichtigen Entscheidung bei der Konfiguration der Identitätsquelle ab: ob Sie ID- oder Zugriffstoken verarbeiten. Bei einem Amazon Cognito Cognito-Identitätsanbieter haben Sie die Wahl zwischen dem Tokentyp, wenn Sie einen API-verknüpften Richtlinienspeicher erstellen. Wenn Sie einen API-verknüpften Richtlinienspeicher erstellen, müssen Sie wählen, ob Sie die Autorisierung für ID- oder Zugriffstoken einrichten möchten. Diese Informationen wirken sich auf die Schemaattribute aus, die Verified Permissions auf Ihren Richtlinienspeicher anwendet, und auf die Syntax des Lambda-Autorisierers für Ihre API-Gateway-API. Bei einem OIDC-Anbieter müssen Sie beim Hinzufügen der Identitätsquelle einen Tokentyp auswählen. Sie können zwischen ID und Zugriffstoken wählen, und Ihre Wahl schließt die Verarbeitung des nicht ausgewählten Tokentyps in Ihrem Richtlinienspeicher aus. Insbesondere, wenn Sie von der automatischen Zuordnung von ID-Token-Ansprüchen zu Attributen in der Verified Permissions-Konsole profitieren möchten, entscheiden Sie sich frühzeitig für den Tokentyp, den Sie verarbeiten möchten, bevor Sie Ihre Identitätsquelle erstellen. Das Ändern des Tokentyps erfordert einen erheblichen Aufwand, um Ihre Richtlinien und Ihr Schema umzugestalten. In den folgenden Themen wird die Verwendung von ID- und Zugriffstoken mit Richtlinienspeichern beschrieben.

Der Cedar-Parser benötigt für einige Zeichen Klammern

Richtlinien verweisen normalerweise auf Schemaattribute in einem Format wie`principal.username`. Bei den meisten nicht-alphanumerischen Zeichen wie`:`, oder`.`,

/ die in Namen von Token-Ansprüchen vorkommen können, kann Verified Permissions einen Bedingungswert wie oder nicht analysieren. `principal.cognito:username` `context.ip-address` Stattdessen müssen Sie diese Bedingungen mit Klammernotation im jeweiligen Format `principal["cognito:username"]` oder `context["ip-address"]` formatieren. Der Unterstrich _ ist ein gültiges Zeichen in Anspruchsnamen und die einzige nicht alphanumerische Ausnahme von dieser Anforderung.

Ein teilweises Beispielschema für ein Hauptattribut dieses Typs sieht wie folgt aus:

```
"User": {
   "shape": {
      "type": "Record",
      "attributes": {
         "cognito:username": {
            "type": "String",
            "required": true
         },
         "custom:employmentStoreCode": {
            "type": "String",
            "required": true,
         },
         "email": {
            "type": "String",
            "required": false
         }
      }
   }
}
```

Ein teilweises Beispielschema für ein Kontextattribut dieses Typs sieht wie folgt aus:

```
"GetOrder": {
   "memberOf": [],
   "appliesTo": {
      "resourceTypes": [
         "Order"
      ],
      "context": {
         "type": "Record",
         "attributes": {
            "ip-address": {
               "required": false,
```

```
                "type": "String"
            }
    }
    },
        "principalTypes": [
            "User"
        ]
    }
}
```

Ein Beispiel für eine Richtlinie, die anhand dieses Schemas validiert wird, finden Sie unterVerwendet die Klammernotation, um auf Token-Attribute zu verweisen.

Integrationen für von Amazon verifizierte Berechtigungen

Die Integrationen von Amazon Verified Permissions helfen Ihnen dabei, eine differenzierte Autorisierung in Ihren Anwendungen zu implementieren und gleichzeitig den Code zu minimieren und Framework-spezifische Best Practices zu befolgen. Diese Integrationen bieten Middleware-Komponenten und Dienstprogramme, die Ihre Anwendung nahtlos mit Verified Permissions verbinden.

Mit Integrationen können Sie:

- Implementieren Sie die Autorisierung in wenigen Minuten
- Halten Sie sich an Framework-spezifische Muster und Konventionen
- Reduzieren Sie den Wartungsaufwand
- Minimieren Sie potenzielle Fehler bei der Sicherheitsimplementierung
- Konzentrieren Sie sich eher auf die Geschäftslogik als auf den Autorisierungscode

Wenn Integrationen zu Ihrer Anwendung hinzugefügt werden, bewirken sie Folgendes:

1. Fangen Sie eingehende Anfragen über Framework-spezifische Middleware ab
2. Extrahieren Sie den relevanten Autorisierungskontext aus Anfragen
3. Ermitteln Sie Autorisierungsentscheidungen mithilfe verifizierter Berechtigungen
4. Erzwingen Sie die Zugriffskontrolle auf der Grundlage der Autorisierungsergebnisse

Verified Permissions unterstützt derzeit die folgenden Frameworks:

- Express.js für Node.js -Anwendungen

Integration von Express mit Amazon Verified Permissions

Die Verified Permissions Express-Integration bietet einen Middleware-basierten Ansatz zur Implementierung der Autorisierung in Ihren Express.js -Anwendungen. Mit dieser Integration können Sie Ihre API-Endpunkte mithilfe detaillierter Autorisierungsrichtlinien schützen, ohne Ihre vorhandenen Route-Handler ändern zu müssen. Die Integration wickelt Autorisierungsprüfungen automatisch ab, indem sie Anfragen abfängt, sie anhand Ihrer definierten Richtlinien bewertet und sicherstellt, dass nur autorisierte Benutzer auf geschützte Ressourcen zugreifen können.

Dieses Thema führt Sie durch die Einrichtung der Express-Integration, von der Erstellung eines Richtlinienspeichers bis hin zur Implementierung und zum Testen der Autorisierungs-Middleware. Wenn Sie diese Schritte befolgen, können Sie Ihrer Express-Anwendung mit minimalen Codeänderungen robuste Autorisierungskontrollen hinzufügen.

In diesem Thema GitHub wird immer wieder auf die folgenden Repos verwiesen:

- cedar-policy/ authorization-for-expressjs — Die Cedar-Autorisierungs-Middleware für Express.js
- verifiedpermissions/ — Die Verified Permissions-Autorisierungsclients für authorization-clients-js JavaScript
- verifiedpermissions/examples/express-petstore — Beispielimplementierung mit der Middleware Express.js

Voraussetzungen

Bevor Sie die Express-Integration implementieren, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein AWS Konto mit Zugriff auf verifizierte Berechtigungen
- Node.js und npm sind installiert
- Eine Express.js -Anwendung
- Ein OpenID Connect (OIDC) -Identitätsanbieter (wie Amazon Cognito)
- AWS CLImit den entsprechenden Berechtigungen konfiguriert

Einrichtung der Integration

Schritt 1: Erstellen Sie einen Richtlinienspeicher

Erstellen Sie einen Richtlinienspeicher mit dem AWS CLI:

```
aws verifiedpermissions create-policy-store --validation-settings "mode=STRICT"
```

> **Note**
>
> Speichern Sie die in der Antwort zurückgegebene Policy Store-ID zur Verwendung in nachfolgenden Schritten.

Schritt 2: Abhängigkeiten installieren

Installieren Sie die erforderlichen Pakete in Ihrer Express-Anwendung:

```
npm i --save @verifiedpermissions/authorization-clients-js
npm i --save @cedar-policy/authorization-for-expressjs
```

Autorisierung konfigurieren

Schritt 1: Generieren Sie das Cedar-Schema und laden Sie es hoch

Ein Schema definiert das Autorisierungsmodell für eine Anwendung, einschließlich der Entitätstypen in der Anwendung und der Aktionen, die Benutzer ausführen dürfen. Wir empfehlen, einen Namespace für Ihr Schema zu definieren. In diesem Beispiel verwenden wir `YourNamespace`. Sie hängen Ihr Schema an Ihre Richtlinienspeicher für verifizierte Berechtigungen an. Wenn Richtlinien hinzugefügt oder geändert werden, validiert der Dienst die Richtlinien automatisch anhand des Schemas.

Das `@cedar-policy/authorization-for-expressjs` Paket kann die OpenAPI-Spezifikationen Ihrer Anwendung analysieren und ein Cedar-Schema generieren. Insbesondere ist das Paths-Objekt in Ihrer Spezifikation erforderlich.

Wenn Sie keine OpenAPI-Spezifikation haben, können Sie den Kurzanweisungen des express-openapi-generatorPakets folgen, um eine OpenAPI-Spezifikation zu generieren.

Generieren Sie ein Schema aus Ihrer OpenAPI-Spezifikation:

```
npx @cedar-policy/authorization-for-expressjs generate-schema --api-spec schemas/
openapi.json --namespace YourNamespace --mapping-type SimpleRest
```

Formatieren Sie als Nächstes das Cedar-Schema für die Verwendung mit dem AWS CLI. Weitere Informationen zu dem jeweils erforderlichen Format finden Sie unterSchema des Richtlinienspeichers. Wenn Sie Hilfe beim Formatieren des Schemas benötigen, gibt es ein Skript, das `prepare-cedar-schema.sh` im Repo GitHubverifiedpermissions/examples aufgerufen wird. Im Folgenden finden Sie einen Beispielaufruf dieses Skripts, das das formatierte Schema Verified Permissions in der Datei ausgibt. `v2.cedarschema.forAVP.json`

```
./scripts/prepare-cedar-schema.sh v2.cedarschema.json v2.cedarschema.forAVP.json
```

Laden Sie das formatierte Schema in Ihren Richtlinienspeicher hoch und `policy-store-id` ersetzen Sie es durch Ihre Policy-Speicher-ID:

```
aws verifiedpermissions put-schema \
  --definition file://v2.cedarschema.forAVP.json \
  --policy-store-id policy-store-id
```

Schritt 2: Autorisierungsrichtlinien erstellen

Wenn keine Richtlinien konfiguriert sind, lehnt Cedar alle Autorisierungsanfragen ab. Die Express-Framework-Integration hilft dabei, diesen Prozess zu beschleunigen, indem sie Beispielrichtlinien generiert, die auf dem zuvor generierten Schema basieren.

Wenn Sie diese Integration in Ihren Produktionsanwendungen verwenden, empfehlen wir, mithilfe von IaaC-Tools (Infrastructure as a Code) neue Richtlinien zu erstellen. Weitere Informationen finden Sie unter Arbeiten mit AWS CloudFormation.

Generieren Sie Cedar-Beispielrichtlinien:

```
npx @cedar-policy/authorization-for-expressjs generate-policies --schema
 v2.cedarschema.json
```

Dadurch werden Beispielrichtlinien im `/policies` Verzeichnis generiert. Sie können diese Richtlinien dann an Ihre Anwendungsfälle anpassen. Zum Beispiel:

```
// Defines permitted administrator user group actions
permit (
    principal in YourNamespace::UserGroup::"<userPoolId>|administrator",
    action,
    resource
);

// Defines permitted employee user group actions
permit (
    principal in YourNamespace::UserGroup::"<userPoolId>|employee",
    action in
        [YourNamespace::Action::"GET /resources",
         YourNamespace::Action::"POST /resources",
         YourNamespace::Action::"GET /resources/{resourceId}",
         YourNamespace::Action::"PUT /resources/{resourceId}"],
    resource
```

```
);
```

Formatieren Sie die Richtlinien für die Verwendung mit dem AWS CLI. Weitere Informationen zum erforderlichen Format finden Sie unter create-policy in der AWS CLI Referenz. Wenn Sie Hilfe beim Formatieren der Richtlinien benötigen, gibt es ein Skript, das `convert_cedar_policies.sh` im Repo GitHubverifiedpermissions/examples aufgerufen wird. Das Folgende ist ein Aufruf dieses Skripts:

```
./scripts/convert_cedar_policies.sh
```

Laden Sie die formatierten Richtlinien in Verified Permissions hoch und `policy_1.json` ersetzen Sie sie durch den Pfad und den Namen Ihrer Richtliniendatei sowie `policy-store-id` durch Ihre Policy-Store-ID:

```
aws verifiedpermissions create-policy \
  --definition file://policies/json/policy_1.json \
  --policy-store-id policy-store-id
```

Schritt 3: Einen Identitätsanbieter verbinden

Standardmäßig liest die Authorizer-Middleware Verified Permissions ein JSON Web Token (JWT), das im Autorisierungsheader der API-Anfrage bereitgestellt wird, um Benutzerinformationen abzurufen. Verified Permissions kann das Token zusätzlich zur Bewertung der Autorisierungsrichtlinien validieren.

Erstellen Sie eine Konfigurationsdatei für `identity-source-configuration.txt` die Identitätsquelle mit dem Namen `userPoolArn` und, die wie folgt aussieht`clientId`:

```
{
    "cognitoUserPoolConfiguration": {
        "userPoolArn": "arn:aws:cognito-idp:region:account:userpool/pool-id",
        "clientIds": ["client-id"],
        "groupConfiguration": {
            "groupEntityType": "YourNamespace::UserGroup"
        }
    }
}
```

Erstellen Sie die Identitätsquelle, indem Sie den folgenden AWS CLI Befehl ausführen und ihn `policy-store-id` durch Ihre Policy Store-ID ersetzen:

```
aws verifiedpermissions create-identity-source \
  --configuration file://identity-source-configuration.txt \
  --policy-store-id policy-store-id \
  --principal-entity-type YourNamespace::User
```

Implementierung der Autorisierungs-Middleware

Aktualisieren Sie Ihre Express-Anwendung so, dass sie die Autorisierungs-Middleware enthält. In diesem Beispiel verwenden wir Identitätstoken, aber Sie können auch Zugriffstoken verwenden. Weitere Informationen finden Sie [authorization-for-expressjs](#)unterGitHub.

```
const { ExpressAuthorizationMiddleware } = require('@cedar-policy/authorization-for-
expressjs');

const { AVPAuthorizationEngine } = require('@verifiedpermissions/authorization-
clients');

const avpAuthorizationEngine = new AVPAuthorizationEngine({
    policyStoreId: 'policy-store-id',
    callType: 'identityToken'
});

const expressAuthorization = new ExpressAuthorizationMiddleware({
    schema: {
        type: 'jsonString',
        schema: fs.readFileSync(path.join(__dirname, '../v4.cedarschema.json'),
 'utf8'),
    },
    authorizationEngine: avpAuthorizationEngine,
    principalConfiguration: { type: 'identityToken' },
    skippedEndpoints: [],
    logger: {
        debug: (s) => console.log(s),
        log: (s) => console.log(s),
    }
});

// Add the middleware to your Express application
app.use(expressAuthorization.middleware);
```

Die Integration testen

Sie können Ihre Autorisierungsimplementierung testen, indem Sie Anfragen mit unterschiedlichen Benutzertoken an Ihre API-Endpunkte stellen. Die Autorisierungs-Middleware bewertet jede Anfrage automatisch anhand Ihrer definierten Richtlinien.

Wenn Sie beispielsweise verschiedene Benutzergruppen mit unterschiedlichen Berechtigungen eingerichtet haben:

- Administratoren: Voller Zugriff auf alle Ressourcen und Verwaltungsfunktionen
- Mitarbeiter: Können Ressourcen anzeigen, erstellen und aktualisieren
- Kunden: Sie können nur Ressourcen anzeigen

Sie können überprüfen, ob die Berechtigungsrichtlinien erwartungsgemäß funktionieren, indem Sie sich mit verschiedenen Benutzern anmelden und verschiedene Vorgänge ausprobieren. Im Terminal für die Express-Anwendung finden Sie eine Protokollausgabe mit zusätzlichen Informationen zu den Autorisierungsentscheidungen.

Fehlerbehebung

Wenn die Autorisierung fehlschlägt, versuchen Sie Folgendes:

- Stellen Sie sicher, dass Ihre Policy Store-ID korrekt ist
- Stellen Sie sicher, dass Ihre Identitätsquelle richtig konfiguriert ist
- Vergewissern Sie sich, dass Ihre Richtlinien korrekt formatiert sind
- Stellen Sie sicher, dass Ihre JWT-Token gültig sind

Nächste Schritte

Nach der Implementierung der Basisintegration sollten Sie Folgendes in Betracht ziehen:

- Implementierung benutzerdefinierter Mapper für bestimmte Autorisierungsszenarien
- Einrichtung der Überwachung und Protokollierung von Autorisierungsentscheidungen
- Erstellung zusätzlicher Richtlinien für verschiedene Benutzerrollen

Implementierung der Autorisierung in Amazon Verified Permissions

Nachdem Sie Ihren Richtlinienspeicher, Ihre Richtlinien, Vorlagen, Ihr Schema und Ihr Autorisierungsmodell erstellt haben, können Sie mit der Autorisierung von Anfragen mithilfe von Amazon Verified Permissions beginnen. Um die Autorisierung mit verifizierten Berechtigungen zu implementieren, müssen Sie die Konfiguration der Autorisierungsrichtlinien AWS mit der Integration in eine Anwendung kombinieren. Um Verified Permissions in Ihre Anwendung zu integrieren, fügen Sie ein AWS SDK hinzu und implementieren Sie die Methoden, die die Verified Permissions API aufrufen und Autorisierungsentscheidungen anhand Ihres Richtlinienspeichers generieren.

Die Autorisierung mit verifizierten Berechtigungen ist nützlich für UX-Berechtigungen und API-Berechtigungen in Ihren Anwendungen.

UX-Berechtigungen

: Steuern Sie den Benutzerzugriff auf Ihre Anwendungs-UX. Sie können einem Benutzer erlauben, nur genau die Formulare, Schaltflächen, Grafiken und anderen Ressourcen anzuzeigen, auf die er zugreifen muss. Wenn sich ein Benutzer beispielsweise anmeldet, möchten Sie vielleicht festlegen, ob die Schaltfläche „Geld überweisen“ in seinem Konto sichtbar ist. Sie können auch die Aktionen steuern, die ein Benutzer ausführen kann. Beispielsweise möchten Sie in derselben Banking-App möglicherweise festlegen, ob Ihr Benutzer die Kategorie einer Transaktion ändern darf.

API-Berechtigungen

: Steuern Sie den Benutzerzugriff auf Daten. Anwendungen sind häufig Teil eines verteilten Systems und beziehen Informationen von außen ein APIs. Im Beispiel der Banking-App, bei der Verified Permissions die Anzeige einer Schaltfläche „Geld überweisen“ zugelassen hat, muss eine komplexere Autorisierungsentscheidung getroffen werden, wenn Ihr Benutzer eine Überweisung veranlasst. Verified Permissions kann die API-Anfrage autorisieren, in der die Zielkonten aufgeführt sind, die als Übertragungsziele in Frage kommen, und dann die Anfrage, die Übertragung auf das andere Konto zu übertragen.

Die Beispiele, die diesen Inhalt veranschaulichen, stammen aus einem Beispielrichtlinienspeicher. Um dem nachzugehen, erstellen Sie den DigitalPetStoreBeispiel-Richtlinienspeicher in Ihrer Testumgebung.

Eine durchgängige Beispielanwendung, die UX-Berechtigungen mithilfe der Batch-Autorisierung implementiert, finden Sie im AWS Sicherheits-Blog unter Verwenden Sie von Amazon verifizierte Berechtigungen für eine detaillierte Autorisierung im großen Maßstab.

Themen

- Verfügbare API-Operationen für die Autorisierung
- Testen Sie Ihr Autorisierungsmodell
- Integrieren Sie Ihre Autorisierungsmodelle mit Anwendungen

Verfügbare API-Operationen für die Autorisierung

Die Verified Permissions API verfügt über die folgenden Autorisierungsvorgänge.

IsAuthorized

Der `IsAuthorized` API-Vorgang ist der Einstiegspunkt für Autorisierungsanfragen mit verifizierten Berechtigungen. Sie müssen die Elemente „Principal“, „Action“, „Resource“, „Context“ und „Entities“ einreichen. Verified Permissions validiert die Entitäten in Ihrer Anfrage anhand Ihres Policy-Store-Schemas. Verified Permissions bewertet dann Ihre Anfrage anhand aller Richtlinien im angeforderten Richtlinienspeicher, die für die Entitäten in der Anfrage gelten.

IsAuthorizedWithToken

Der `IsAuthorizedWithToken` Vorgang generiert eine Autorisierungsanfrage anhand von Benutzerdaten in JSON-Webtoken (JWTs). Verified Permissions arbeitet direkt mit OIDC-Anbietern wie Amazon Cognito als Identitätsquelle in Ihrem Richtlinienspeicher zusammen. Verified Permissions füllt alle Attribute für den Principal in Ihrer Anfrage anhand der Ansprüche in der Benutzer-ID oder in den Zugriffstoken auf. Sie können Aktionen und Ressourcen anhand von Benutzerattributen oder der Gruppenmitgliedschaft in einer Identitätsquelle autorisieren.

Sie können keine Informationen über Gruppen- oder Benutzerprinzipaltypen in eine `IsAuthorizedWithToken` Anfrage aufnehmen. Sie müssen alle Prinzipaldaten in das von Ihnen bereitgestellte JWT eingeben.

BatchIsAuthorized

Der `BatchIsAuthorized` Vorgang verarbeitet mehrere Autorisierungsentscheidungen für einen einzelnen Prinzipal oder eine einzelne Ressource in einer einzigen API-Anforderung. Dieser Vorgang gruppiert Anfragen in einem einzigen Batch-Vorgang, der die Quotennutzung minimiert

und Autorisierungsentscheidungen für jede der bis zu 30 komplexen verschachtelten Aktionen zurückgibt. Mit der Batch-Autorisierung für eine einzelne Ressource können Sie die Aktionen filtern, die ein Benutzer für eine Ressource ausführen kann. Mit der Batch-Autorisierung für einen einzelnen Prinzipal können Sie nach den Ressourcen filtern, für die ein Benutzer Aktionen ausführen kann.

BatchIsAuthorizedWithToken

Der `BatchIsAuthorizedWithToken` Vorgang verarbeitet mehrere Autorisierungsentscheidungen für einen einzelnen Prinzipal in einer API-Anfrage. Der Prinzipal wird von der Identitätsquelle Ihres Richtlinienspeichers in einer ID oder einem Zugriffstoken bereitgestellt. Dieser Vorgang gruppiert Anfragen in einem einzigen Batch-Vorgang, der die Kontingentnutzung minimiert und Autorisierungsentscheidungen für jede von bis zu 30 Anfragen nach Aktionen und Ressourcen zurückgibt. In Ihren Richtlinien können Sie ihren Zugriff über ihre Attribute oder ihre Gruppenmitgliedschaft in einem Benutzerverzeichnis autorisieren.

Wie bei `IsAuthorizedWithToken` können Sie keine Informationen über Gruppen- oder Benutzerprinzipaltypen in eine `BatchIsAuthorizedWithToken` Anfrage aufnehmen. Sie müssen alle Prinzipaldaten in das von Ihnen bereitgestellte JWT eingeben.

Testen Sie Ihr Autorisierungsmodell

Um zu verstehen, wie sich die Autorisierungsentscheidung von Amazon Verified Permissions bei der Bereitstellung Ihrer Anwendung auswirkt, können Sie Ihre Richtlinien bei der Entwicklung mit den Verwenden des Amazon Verified Permissions-Testbench und mit HTTPS-REST-API-Anfragen an Verified Permissions evaluieren. Der Prüfstand ist ein Tool AWS Management Console zur Bewertung von Autorisierungsanfragen und Antworten in Ihrem Richtlinienspeicher.

Die REST-API für verifizierte Berechtigungen ist der nächste Schritt in Ihrer Entwicklung, wenn Sie von einem konzeptionellen Verständnis zum Anwendungsdesign übergehen. Die Verified Permissions API akzeptiert Autorisierungsanfragen mit IsAuthorizedIsAuthorizedWithToken, und BatchIsAuthorizedals signierte AWS API-Anfragen an regionale Service-Endpunkte. Um Ihr Autorisierungsmodell zu testen, können Sie Anfragen mit einem beliebigen API-Client generieren und überprüfen, ob Ihre Richtlinien die Autorisierungsentscheidungen erwartungsgemäß zurückgeben.

Mit dem folgenden Verfahren können Sie beispielsweise `IsAuthorized` in einem Beispiel-Richtlinienspeicher testen.

Test bench

1. Öffnen Sie die Konsole Verified Permissions unter Verified Permissions console. Erstellen Sie aus dem Beispielrichtlinienspeicher einen Richtlinienspeicher mit dem Namen DigitalPetStore.
2. Wählen Sie in Ihrem neuen Richtlinienspeicher die Option Testbench aus.
3. Füllen Sie Ihre Testbench-Anfrage IsAuthorizedin der API-Referenz für verifizierte Berechtigungen aus. Die folgenden Details entsprechen den Bedingungen in Beispiel 4, das auf das DigitalPetStoreBeispiel verweist.

    a. Stellen Sie Alice als Principal ein. Wählen Sie für Principal taking action die Option `DigitalPetStore::User` und geben Sie ein`Alice`.

    b. Lege Alices Rolle als Kunde fest. Wählen Sie Elternteil hinzufügen`DigitalPetStore::Role`, wählen Sie und geben Sie Kunde ein.

    c. Geben Sie für die Ressource die Bestellung „1234" ein. Wählen Sie unter Ressource, auf die der Principal reagiert, die Option aus `DigitalPetStore::Order` und geben Sie ein`1234`.

    d. Die `DigitalPetStore::Order` Ressource benötigt ein `owner` Attribut. Lege Alice als Eigentümerin der Bestellung fest. Wähle `DigitalPetStore::User` und gib ein `Alice`

    e. Alice bat darum, die Bestellung einzusehen. Wählen Sie für Aktion, die der Schulleiter gerade ergreift, die Option`DigitalPetStore::Action::"GetOrder"`.
4. Wählen Sie Autorisierungsanfrage ausführen aus. In einem unveränderten Richtlinienspeicher führt diese Anfrage zu einer `ALLOW` Entscheidung. Notieren Sie sich die Richtlinie „Zufrieden“, die die Entscheidung zurückgegeben hat.
5. Wählen Sie in der linken Navigationsleiste Richtlinien aus. Überprüfen Sie die statische Richtlinie mit der Beschreibung Kundenrolle — Bestellung abrufen.
6. Beachten Sie, dass Verified Permissions die Anfrage zugelassen hat, weil der Principal eine Kundenrolle innehatte und der Eigentümer der Ressource war.

REST API

1. Öffnen Sie die Konsole Verified Permissions unter Verified Permissions console. Erstellen Sie aus dem Beispielrichtlinienspeicher einen Richtlinienspeicher mit dem Namen DigitalPetStore.
2. Notieren Sie sich die Policy-Store-ID Ihres neuen Policy-Speichers.

3. Kopieren Sie aus IsAuthorizedder API-Referenz für verifizierte Berechtigungen den Anfragetext von Beispiel 4, der auf das DigitalPetStoreBeispiel verweist.
4. Öffnen Sie Ihren API-Client und erstellen Sie eine Anfrage an den regionalen Service-Endpunkt für Ihren Richtlinienspeicher. Füllen Sie die Header wie im Beispiel gezeigt aus.
5. Fügen Sie den Text der Beispielanforderung ein und ändern Sie den Wert von in `policyStoreId` die zuvor notierte Policy Store-ID.
6. Reichen Sie die Anfrage ein und überprüfen Sie die Ergebnisse. In einem DigitalPetStoreStandardrichtlinienspeicher gibt diese Anfrage eine `ALLOW` Entscheidung zurück.

Sie können in Ihrer Testumgebung Änderungen an Richtlinien, Schemas und Anforderungen vornehmen, um die Ergebnisse zu ändern und komplexere Entscheidungen zu treffen.

1. Ändern Sie die Anfrage so, dass die Entscheidung unter Verifizierte Berechtigungen geändert wird. Ändern Sie beispielsweise Alices Rolle auf `Employee` oder ändern Sie das `owner` Attribut der Bestellung 1234 auf`Bob`.
2. Ändern Sie Richtlinien so, dass sie sich auf Autorisierungsentscheidungen auswirken. Ändern Sie beispielsweise die Richtlinie mit der Beschreibung Kundenrolle — Bestellung abrufen, um die Bedingung zu entfernen, dass der Eigentümer der Anfrage sein `User` muss, `Resource` und ändern Sie die Anfrage so, dass der Kunde die Bestellung einsehen `Bob` möchte.
3. Ändern Sie das Schema, damit Richtlinien komplexere Entscheidungen treffen können. Aktualisieren Sie die Anforderungsentitäten, damit Alice die neuen Anforderungen erfüllen kann. Bearbeiten Sie das Schema beispielsweise so, dass `User` Sie Mitglied von `ActiveUsers` oder sein können`InactiveUsers`. Aktualisieren Sie die Richtlinie, sodass nur aktive Benutzer ihre eigenen Bestellungen einsehen können. Aktualisieren Sie die Anforderungsentitäten, sodass Alice ein aktiver oder inaktiver Benutzer ist.

Integrieren Sie Ihre Autorisierungsmodelle mit Anwendungen

Um Amazon Verified Permissions in Ihrer Anwendung zu implementieren, müssen Sie die Richtlinien und das Schema definieren, die Ihre App durchsetzen soll. Nachdem Ihr Autorisierungsmodell eingerichtet und getestet wurde, besteht Ihr nächster Schritt darin, API-Anfragen vom Punkt der Durchsetzung aus zu generieren. Dazu müssen Sie eine Anwendungslogik einrichten, um Benutzerdaten zu sammeln und diese für Autorisierungsanfragen zu verwenden.

Wie eine App Anfragen mit verifizierten Berechtigungen autorisiert

1. Sammeln Sie Informationen über den aktuellen Benutzer. In der Regel werden die Details eines Benutzers in den Details einer authentifizierten Sitzung bereitgestellt, z. B. in einem JWT- oder Websitzungs-Cookie. Diese Benutzerdaten können aus einer Amazon Cognito Cognito-Identitätsquelle stammen, die mit Ihrem Policy Store verknüpft ist, oder von einem anderen OpenID Connect (OIDC) -Anbieter.
2. Sammeln Sie Informationen über die Ressource, auf die ein Benutzer zugreifen möchte. In der Regel erhält Ihre Anwendung Informationen über die Ressource, wenn ein Benutzer eine Auswahl trifft, sodass Ihre App ein neues Asset laden muss.
3. Bestimmen Sie die Aktion, die Ihr Benutzer ergreifen möchte.
4. Generieren Sie eine Autorisierungsanfrage an Verified Permissions mit dem Principal, der Aktion, der Ressource und den Entitäten für den versuchten Vorgang Ihres Benutzers. Verified Permissions bewertet die Anfrage anhand der Richtlinien in Ihrem Richtlinienspeicher und gibt eine Autorisierungsentscheidung zurück.
5. Ihre Anwendung liest die Antwort „Zulassen“ oder „Verweigern“ von Verified Permissions und erzwingt die Entscheidung in Bezug auf die Anfrage des Benutzers.

API-Operationen für verifizierte Berechtigungen sind integriert AWS SDKs. Um verifizierte Berechtigungen in eine App aufzunehmen, integrieren Sie das AWS SDK für die von Ihnen gewählte Sprache in das App-Paket.

Weitere Informationen und Downloads AWS SDKs finden Sie unter Tools für Amazon Web Services.

Im Folgenden finden Sie Links zur Dokumentation für Ressourcen zu verifizierten Berechtigungen in verschiedenen Bereichen AWS SDKs.

- AWS SDK für .NET
- AWS SDK für C++
- AWS SDK für Go
- AWS SDK für Java
- AWS SDK für JavaScript
- AWS SDK für PHP
- AWS SDK for Python (Boto)
- AWS SDK für Ruby

- AWS SDK for Rust

Das folgende AWS SDK für JavaScript Beispiel für `IsAuthorized` stammt aus der detaillierten Autorisierung von Simplify mit Amazon Verified Permissions und Amazon Cognito.

```
const authResult = await avp.isAuthorized({
    principal: 'User::"alice"',
    action: 'Action::"view"',
    resource: 'Photo::"VacationPhoto94.jpg"',
    // whenever our policy references attributes of the entity,
    // isAuthorized needs an entity argument that provides
    // those attributes
    entities: {
       entityList: [
         {
            "identifier": {
                "entityType": "User",
                "entityId": "alice"
            },
            "attributes": {
                "location": {
                    "String": "USA"
                }
            }
         }
       ]
    }
});
```

Weitere Ressourcen für Entwickler

- Workshop „Von Amazon verifizierte Berechtigungen“
- Von Amazon verifizierte Berechtigungen — Ressourcen
- Implementieren Sie einen benutzerdefinierten Autorisierungsrichtlinienanbieter für ASP.NET Core-Apps mithilfe von Amazon Verified Permissions
- Erstellen Sie mithilfe von Amazon Verified Permissions einen Berechtigungsservice für Geschäftsanwendungen
- Vereinfachen Sie die detaillierte Autorisierung mit Amazon Verified Permissions und Amazon Cognito

Sicherheit bei von Amazon verifizierten Berechtigungen

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, auf der AWS Dienste in der ausgeführt AWS Cloud werden. 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 Amazon Verified Permissions gelten, finden Sie unter AWS Services in Scope by Compliance Program AWS .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung verifizierter Berechtigungen anwenden können. In den folgenden Themen erfahren Sie, wie Sie verifizierte Berechtigungen konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Außerdem erfahren Sie, wie Sie andere AWS Dienste nutzen können, mit denen Sie Ihre Ressourcen für verifizierte Berechtigungen überwachen und schützen können.

Themen

- Datenschutz in Amazon Verified Permissions
- Identitäts- und Zugriffsmanagement für Amazon Verified Permissions
- Konformitätsprüfung für von Amazon Verified Permissions
- Resilienz bei von Amazon verifizierten Berechtigungen

Datenschutz in Amazon Verified Permissions

Das AWS Modell der gilt für den Datenschutz in Amazon Verified Permissions. Wie in diesem Modell beschrieben, ist AWS für den Schutz der globalen Infrastruktur verantwortlich, in der die

gesamte AWS Cloud ausgeführt wird. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Dieser Inhalt umfasst die Sicherheitskonfiguration und die Verwaltungsaufgaben für AWS-Services das, was Sie verwenden. 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.
- Wir empfehlen Ihnen, Ihre Daten auf folgende Weise zu sichern:
  - Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
  - Verwenden Sie SSL/TLS, um mit AWS Ressourcen zu kommunizieren. Wir erfordern TLS 1.2.
  - 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 auf AWS über eine Befehlszeilenschnittstelle oder über 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. Dazu gehört auch, wenn Sie mit verifizierten Berechtigungen oder auf andere Weise AWS-Services über die Konsole, die API oder arbeiten AWS SDKs. AWS CLI 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.
- Ihre Aktionsnamen sollten keine vertraulichen Informationen enthalten.
- Wir empfehlen außerdem dringend, immer eindeutige, nicht veränderbare und nicht wiederverwendbare Identifikatoren für Ihre Entitäten (Ressourcen und Prinzipale) zu verwenden. In einer Testumgebung können Sie sich dafür entscheiden, einfache Entitätsbezeichner wie `jane`

oder bob für den Namen einer Entität des Typs zu verwenden. `User` In einem Produktionssystem ist es jedoch aus Sicherheitsgründen wichtig, dass Sie eindeutige Werte verwenden, die nicht wiederverwendet werden können. Wir empfehlen, Werte wie Universally Unique Identifiers () zu verwenden. UUIDs Denken Sie beispielsweise an den Benutzer`jane`, der das Unternehmen verlässt. Später lassen Sie jemand anderen den Namen verwenden`jane`. Dieser neue Benutzer erhält automatisch Zugriff auf alles, was durch Richtlinien gewährt wird, auf die immer noch verwiesen wird`User::"jane"`. Verifizierte Berechtigungen und Cedar können nicht zwischen dem neuen Benutzer und dem vorherigen Benutzer unterscheiden.

Diese Anleitung gilt sowohl für Haupt- als auch für Ressourcen-IDs. Verwenden Sie immer Identifikatoren, die garantiert einzigartig sind und niemals wiederverwendet werden, um sicherzustellen, dass Sie nicht versehentlich Zugriff gewähren, weil eine Richtlinie eine alte Kennung enthält.

- Stellen Sie sicher, dass die Zeichenketten, die Sie zur Definition angeben, `Long` und die `Decimal` Werte innerhalb des gültigen Bereichs für jeden Typ liegen. Stellen Sie außerdem sicher, dass Ihre Verwendung von arithmetischen Operatoren nicht zu einem Wert außerhalb des gültigen Bereichs führt. Wenn der Bereich überschritten wird, führt der Vorgang zu einer Überlaufausnahme. Eine Richtlinie, die zu einem Fehler führt, wird ignoriert, was bedeutet, dass eine Genehmigungsrichtlinie den Zugriff möglicherweise unerwartet nicht zulässt oder dass eine Sperrrichtlinie den Zugriff möglicherweise unerwartet nicht blockiert.

Datenverschlüsselung

Amazon Verified Permissions verschlüsselt automatisch alle Kundendaten wie Policen mit einem Von AWS verwalteter Schlüssel, sodass die Verwendung eines vom Kunden verwalteten Schlüssels weder erforderlich noch unterstützt wird.

Identitäts- und Zugriffsmanagement für Amazon Verified Permissions

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf AWS Ressourcen sicher zu kontrollieren. IAM Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Ressourcen mit verifizierten Berechtigungen zu verwenden. IAM ist eine AWS-Service , die Sie ohne zusätzliche Kosten verwenden können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- So funktioniert Amazon Verified Permissions mit IAM
- IAM Richtlinien für verifizierte Berechtigungen
- Beispiele für identitätsbasierte Richtlinien für Amazon Verified Permissions
- AWS verwaltete Richtlinien für Amazon Verified Permissions
- Fehlerbehebung bei Identität und Zugriff auf Amazon Verified Permissions

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt davon ab, welche Arbeit Sie im Bereich Verifizierte Berechtigungen ausführen.

Dienstbenutzer — Wenn Sie den Dienst Verified Permissions für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Wenn Sie für Ihre Arbeit mehr Funktionen von Verified Permissions verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern müssen. Wenn Sie unter Verifizierte Berechtigungen nicht auf eine Funktion zugreifen können, finden Sie weitere Informationen unterFehlerbehebung bei Identität und Zugriff auf Amazon Verified Permissions.

Dienstadministrator — Wenn Sie in Ihrem Unternehmen für die Ressourcen mit verifizierten Berechtigungen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf verifizierte Berechtigungen. Es ist Ihre Aufgabe, zu bestimmen, auf welche Funktionen und Ressourcen von Verified Permissions Ihre Servicebenutzer zugreifen sollen. Anschließend müssen Sie Anfragen an Ihren IAM Administrator senden, um die Berechtigungen Ihrer Servicebenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die grundlegenden Konzepte von zu verstehen IAM. Weitere Informationen darüber, wie Ihr Unternehmen verifizierte Berechtigungen nutzen IAM kann, finden Sie unterSo funktioniert Amazon Verified Permissions mit IAM.

IAM Administrator — Wenn Sie ein IAM Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien zur Verwaltung des Zugriffs auf verifizierte Berechtigungen verfassen können. Beispiele für identitätsbasierte Richtlinien für verifizierte Berechtigungen, die Sie in

verwenden können IAM, finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Amazon Verified Permissions

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine Rolle übernehmen. Root-Benutzer des AWS-Kontos IAM

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 föderierte Identität anmelden, hat Ihr Administrator zuvor einen Identitätsverbund mithilfe von Rollen eingerichtet. IAM Wenn Sie AWS mithilfe eines Verbunds darauf zugreifen, 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 unter AWS Signature Version 4 für API-Anfragen im IAM Benutzerhandbuch.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen bereitstellen. 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 AWS Multi-Faktor-Authentifizierung IAM im Benutzerhandbuch.IAM

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und

dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen. Verwenden Sie diese 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 im Benutzerhandbuch unter Aufgaben, für die Root-Benutzeranmeldedaten erforderlich sind.IAM

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie im Benutzerhandbuch unter Regelmäßiges Wechseln der Zugriffsschlüssel für Anwendungsfälle, für die IAM langfristige Anmeldeinformationen erforderlich sind.

Eine IAM Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern spezifiziert. 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 IAM Admins 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 im Benutzerhandbuch unter Anwendungsfälle für IAMIAM Benutzer.

IAM Rollen

Eine IAM Rolle ist eine Identität innerhalb von Ihnen AWS-Konto , für die bestimmte Berechtigungen gelten. Sie ist einem IAM-Benutzer vergleichbar, jedoch 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 zur Verwendung von Rollen finden Sie unter Verwenden von IAM Rollen im IAM Benutzerhandbuch.

IAM Rollen mit temporären Anmeldeinformationen sind in den folgenden Situationen nützlich:

- 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 im IAM Benutzerhandbuch unter Erstellen einer Rolle für einen externen Identitätsanbieter (Federation). Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Um zu steuern, 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 IAM-Rolle kann eine IAM Rolle übernehmen, um vorübergehend verschiedene Berechtigungen für eine bestimmte Aufgabe zu übernehmen.
- Kontoübergreifender Zugriff: Sie können eine IAM -Rolle verwenden, um jemandem (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 zum Unterschied zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie im Benutzerhandbuch unter Unterschiede zwischen IAM Rollen und ressourcenbasierten Richtlinien.IAM

- Anwendungen, die auf einer Instanz ausgeführt werden Amazon EC2— 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 Dies ist dem Speichern von Zugriffsschlüsseln innerhalb der EC2 Instanz vorzuziehen. Um einer EC2 Instanz eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instanzprofil, das an die Instanz angehängt ist. Ein Instanzprofil enthält die Rolle und ermöglicht Programmen, die auf der EC2 Instanz ausgeführt werden, temporäre Anmeldeinformationen abzurufen. Weitere Informationen finden Sie im Benutzerhandbuch unter Verwenden einer IAM Rolle, um Anwendungen, die IAM auf Amazon EC2 Instanzen ausgeführt werden, Berechtigungen zu gewähren.

Informationen darüber, ob Sie IAM Rollen oder IAM-Benutzer verwenden sollten, finden Sie im Benutzerhandbuch unter Wann sollte eine IAM Rolle (anstelle eines Benutzers) erstellt werden.IAM

Verwalten des Zugriffs mit Richtlinien

Sie steuern den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Die Berechtigungen in den Richtlinien legen fest, ob eine Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zur Struktur und zum Inhalt von JSON-Richtliniendokumenten finden Sie im IAM Benutzerhandbuch unter Überblick über JSON-Richtlinien.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Um Benutzern die Erlaubnis zu erteilen, Aktionen mit den Ressourcen durchzuführen, die sie benötigen, kann ein IAM Administrator IAM Richtlinien erstellen. Der Administrator kann dann die IAM Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen übernehmen.

IAM Richtlinien definieren Berechtigungen für eine Aktion, unabhängig von der Methode, mit der Sie den Vorgang ausführen. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen einer identitätsbasierten Richtlinie finden Sie im IAM Benutzerhandbuch unter Definieren benutzerdefinierter IAM Berechtigungen mit vom Kunden verwalteten Richtlinien.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können. AWS-Konto Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie oder einer Inline-Richtlinie wählen können, finden Sie im IAM Benutzerhandbuch unter Wählen Sie zwischen verwalteten Richtlinien und Inline-Richtlinien.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM Rollenvertrauensrichtlinien und Amazon S3 S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien nicht IAM in einer ressourcenbasierten Richtlinie verwenden.

Zugriffskontrolllisten () ACLs

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Dienste, die Unterstützung ACLs bieten. AWS WAF Weitere Informationen finden Sie unter Übersicht über ACLs die Zugriffskontrollliste (ACL) im Amazon Simple Storage Service Developer Guide.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen — Eine Berechtigungsgrenze ist eine erweiterte Funktion, mit der Sie die maximalen Berechtigungen festlegen, die eine identitätsbasierte Richtlinie einer IAM Entität (IAM-Benutzer oder Rolle) gewähren kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen zu Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM Entitäten im IAM Benutzerhandbuch.
- Dienststeuerungsrichtlinien (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen AWS Organizations. AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie alle Funktionen in einer Organisation aktivieren, können Sie Richtlinien zur Servicesteuerung (SCPs) auf einige oder alle Ihre Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Root-Benutzer des AWS-Kontos Entitäten. Weitere Informationen zu Organizations und SCPs finden Sie unter Richtlinien zur Servicesteuerung im AWS Organizations Benutzerhandbuch.
- Richtlinien zur Ressourcenkontrolle (RCPs) — RCPs sind JSON-Richtlinien, mit denen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten festlegen können, ohne die IAM Richtlinien aktualisieren zu müssen, die jeder Ressource zugeordnet sind, deren Eigentümer Sie sind. Das RCP schränkt die Berechtigungen für Ressourcen in Mitgliedskonten ein und kann

sich auf die effektiven Berechtigungen für Identitäten auswirken, einschließlich der Root-Benutzer des AWS-Kontos, unabhängig davon, ob sie zu Ihrer Organisation gehören. Weitere Informationen zu Organizations RCPs, einschließlich einer Liste AWS-Services dieser Support-Leistungen RCPs, finden Sie unter Resource Control Policies (RCPs) im AWS Organizations Benutzerhandbuch.

- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im Benutzerhandbuch zu IAM .

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM Benutzerhandbuch unter Bewertungslogik für Richtlinien.

So funktioniert Amazon Verified Permissions mit IAM

Bevor Sie IAM den Zugriff auf verifizierte Berechtigungen verwalten, sollten Sie sich darüber informieren, welche IAM Funktionen mit verifizierten Berechtigungen verwendet werden können.

IAM Funktionen, die Sie mit Amazon Verified Permissions verwenden können

IAM Funktion	Unterstützung für verifizierte Berechtigungen
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Bedingungsschlüssel für die Richtlinie	Nein

IAM Funktion	Unterstützung für verifizierte Berechtigungen
ACLs	Nein
ABAC (Tags in Richtlinien)	Ja
Temporäre Anmeldeinformationen	Ja
Prinzipalberechtigungen	Ja
Servicerollen	Nein
Serviceverknüpfte Rollen	Nein

Einen allgemeinen Überblick darüber, wie verifizierte Berechtigungen und andere AWS Dienste mit den meisten IAM Funktionen funktionieren, finden Sie IAM im IAM Benutzerhandbuch unter AWS Dienste, die mit funktionieren.

Identitätsbasierte Richtlinien für verifizierte Berechtigungen

Unterstützt Richtlinien auf Identitätsbasis.	Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen einer identitätsbasierten Richtlinie finden Sie im Benutzerhandbuch unter Definieren benutzerdefinierter IAM Berechtigungen mit vom Kunden verwalteten Richtlinien.IAM

Mit IAM identitätsbasierten Richtlinien können Sie zulässige oder verweigerte Aktionen und Ressourcen sowie die Bedingungen angeben, unter denen Aktionen zulässig oder verweigert werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Weitere Informationen zu allen Elementen, die Sie in einer JSON-Richtlinie verwenden können, finden Sie im IAM Benutzerhandbuch unter Referenz zu IAM JSON-Richtlinienelementen.

Beispiele für identitätsbasierte Richtlinien für verifizierte Berechtigungen

Beispiele für identitätsbasierte Richtlinien für verifizierte Berechtigungen finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Amazon Verified Permissions

Ressourcenbasierte Richtlinien innerhalb von Verified Permissions

Unterstützt ressourcenbasierte Richtlinien	Nein

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM Rollenvertrauensrichtlinien und Amazon S3 S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um den kontoübergreifenden Zugriff zu ermöglichen, können Sie in einer ressourcenbasierten Richtlinie ein ganzes Konto oder IAM Entitäten in einem anderen Konto als Prinzipal angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource unterscheiden AWS-Konten, muss ein IAM Administrator des vertrauenswürdigen Kontos auch der Prinzipalentität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource gewähren. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie IAM im IAM Benutzerhandbuch unter Kontoübergreifender Ressourcenzugriff.

Richtlinienaktionen für verifizierte Berechtigungen

Unterstützt Richtlinienaktionen	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der Aktionen mit verifizierten Berechtigungen finden Sie unter Von Amazon Verified Permissions definierte Aktionen in der Service Authorization Reference.

Bei Richtlinienaktionen unter Verifizierte Berechtigungen wird vor der Aktion das folgende Präfix verwendet:

```
verifiedpermissions
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "verifiedpermissions:action1",
      "verifiedpermissions:action2"
         ]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Get` beginnen, einschließlich der folgenden Aktion:

```
"Action": "verifiedpermissions:Get*"
```

Beispiele für identitätsbasierte Richtlinien für verifizierte Berechtigungen finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Amazon Verified Permissions

Richtlinienressourcen für verifizierte Berechtigungen

Unterstützt Richtlinienressourcen	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der Ressourcentypen mit verifizierten Berechtigungen und deren ARNs Eigenschaften finden Sie unter Ressourcentypen definiert durch Amazon Verified Permissions in der Service Authorization Reference. Informationen darüber, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter Von Amazon Verified Permissions definierte Aktionen.

Bedingungsschlüssel für Richtlinien für verifizierte Berechtigungen

Unterstützt servicespezifische Richtlini enbedingungsschlüssel	Nein

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition` block) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen `AND`-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, AWS wertet die

Bedingung mithilfe einer logischen `OR` Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie im IAM Benutzerhandbuch unter IAM Richtlinienelemente: Variablen und Tags.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontext-Schlüssel für AWS globale Bedingungen im IAM Benutzerhandbuch.

ACLs in Verifizierte Berechtigungen

Unterstützt ACLs	Nein

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

ABAC mit verifizierten Berechtigungen

Unterstützt ABAC (Tags in Richtlinien)	Ja

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In AWS werden diese Attribute als Tags bezeichnet. Sie können Tags an IAM Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Definieren von Berechtigungen mit ABAC-Autorisierung im IAM Benutzerhandbuch. Ein Tutorial mit Schritten zur Einrichtung von ABAC finden Sie im Benutzerhandbuch unter Verwenden der attributebasierten Zugriffskontrolle (ABAC).IAM

Verwendung temporärer Anmeldeinformationen mit verifizierten Berechtigungen

Unterstützt temporäre Anmeldeinformationen	Ja

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen darüber, AWS-Services wie Sie mit temporären Anmeldeinformationen arbeiten können AWS-Services , finden Sie IAM im IAM Benutzerhandbuch unter Informationen zum Arbeiten mit.

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Kennwort anmelden. Wenn Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Rollenwechsel finden Sie im Benutzerhandbuch unter Von einem Benutzer zu einer IAM Rolle (Konsole) wechseln.IAM

Mithilfe der AWS API AWS CLI oder können Sie temporäre Anmeldeinformationen manuell erstellen. Sie können diese temporären Anmeldeinformationen dann für den Zugriff verwenden AWS. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen in IAM.

Serviceübergreifende Prinzipalberechtigungen für verifizierte Berechtigungen

Unterstützt Prinzipal-Berechtigungen	Ja

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion

in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für verifizierte Berechtigungen

Unterstützt Servicerollen	Nein

Eine Servicerolle ist eine IAM Rolle, die ein Dienst übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM Administrator kann eine Servicerolle von innen heraus erstellen, ändern und löschen IAM. Weitere Informationen finden Sie im IAM Benutzerhandbuch unter Erstellen einer Rolle zum Delegieren von Berechtigungen AWS-Service an eine.

Mit Diensten verknüpfte Rollen für verifizierte Berechtigungen

Unterstützt serviceverknüpfte Rollen	Nein

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 dienstbezogene Rollen anzeigen, aber nicht bearbeiten.

Einzelheiten zum Erstellen oder Verwalten von dienstbezogenen Rollen finden Sie unter AWS Dienste, die mit funktionieren. IAM Suchen Sie in der Tabelle nach einem Service mit einem `Yes` in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

IAM Richtlinien für verifizierte Berechtigungen

Verified Permissions verwaltet die Berechtigungen von Benutzern innerhalb Ihrer Anwendung. Damit Ihre Anwendung die verifizierten Berechtigungen aufrufen kann APIs oder damit AWS Management Console Benutzer die Cedar-Richtlinien in einem Richtlinienspeicher für verifizierte Berechtigungen verwalten können, müssen Sie die erforderlichen IAM Berechtigungen hinzufügen.

Identitätsbasierte Richtlinien sind Richtliniendokumente für JSON-Berechtigungen, die Sie an eine Identität anhängen können, z. B. an einen IAM Benutzer, eine Benutzergruppe oder eine Rolle. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen einer identitätsbasierten Richtlinie finden Sie unter IAM Richtlinien erstellen im Benutzerhandbuch. IAM

Mit IAM identitätsbasierten Richtlinien können Sie zulässige oder verweigerte Aktionen und Ressourcen sowie die Bedingungen angeben, unter denen Aktionen zugelassen oder verweigert werden (siehe unten). Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Weitere Informationen zu allen Elementen, die Sie in einer JSON-Richtlinie verwenden können, finden Sie im IAM Benutzerhandbuch unter Referenz zu IAM JSON-Richtlinienelementen.

Action (Aktion)	Beschreibung
CreateIdentitySource	Aktion zum Erstellen einer neuen Identität squelle.
CreatePolicy	Aktion zum Erstellen einer Cedar-Richtlinie in einem Richtlinienspeicher. Sie können entweder eine statische Richtlinie oder eine mit einer Richtlinienvorlage verknüpfte Richtlinie erstellen.
CreatePolicyStore	Aktion zum Erstellen eines neuen Richtlini enspeichers.
CreatePolicyTemplate	Aktion zum Erstellen einer neuen Richtlini envorlage.
DeleteIdentitySource	Aktion zum Löschen einer Identitätsquelle.
DeletePolicy	Aktion zum Löschen einer Richtlinie aus einem Richtlinienspeicher.
DeletePolicyStore	Aktion zum Löschen eines Richtlinienspeichers.
DeletePolicyTemplate	Aktion zum Löschen einer Richtlinienvorlage.

Action (Aktion)	Beschreibung
GetIdentitySource	Aktion zum Abrufen einer Identitätsquelle.
GetPolicy	Aktion zum Abrufen von Informationen zu einer bestimmten Richtlinie.
GetPolicyStore	Aktion zum Abrufen von Informationen über einen bestimmten Richtlinienspeicher.
GetPolicyTemplate	Aktion zum Abrufen einer Richtlinienvorlage.
GetSchema	Aktion zum Abrufen eines Schemas.
IsAuthorized	Aktion zum Abrufen einer Autorisierungsantwort auf der Grundlage der in der Autorisierungsanfrage beschriebenen Parameter.
IsAuthorizedWithToken	Aktion zum Abrufen einer Autorisierungsantwort auf der Grundlage der in der Autorisierungsanfrage beschriebenen Parameter, wobei der Prinzipal aus einem Identitätstoken stammt.
ListIdentitySources	Aktion zum Auflisten aller Identitätsquellen in der AWS-Konto.
ListPolicies	Aktion zum Auflisten aller Richtlinien in einem Richtlinienspeicher.
ListPolicyStores	Aktion zum Auflisten aller Richtlinienspeicher in der AWS-Konto.
ListPolicyTemplates	Aktion zum Auflisten aller Richtlinienvorlagen in der AWS-Konto.
ListTagsForResource	Aktion zum Auflisten aller Tags für eine Ressource.
PutSchema	Aktion zum Hinzufügen eines Schemas zu einem Richtlinienspeicher.

Action (Aktion)	Beschreibung
TagResource	Aktion zum Hinzufügen eines Tags zu einer Ressource.
UpdateIdentitySource	Aktion zum Aktualisieren einer Identitätsquelle.
UpdatePolicy	Aktion zum Aktualisieren einer Richtlinie in einem Richtlinienspeicher.
UpdatePolicyStore	Aktion zum Aktualisieren eines Richtlini enspeichers.
UpdatePolicyTemplate	Aktion zum Aktualisieren einer Richtlini envorlage.
UntagResource	Aktion zum Entfernen eines Tags aus einer Ressource.

IAM Beispielrichtlinie für die Genehmigung der CreatePolicy Aktion:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "verifiedpermissions:CreatePolicy"
            ],
            "Resource": "*"
        }
    ]
 }
```

Beispiele für identitätsbasierte Richtlinien für Amazon Verified Permissions

Standardmäßig sind Benutzer und Rollen nicht berechtigt, Ressourcen mit verifizierten Berechtigungen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder AWS API ausführen. Ein IAM Administrator muss IAM Richtlinien erstellen, die Benutzern und Rollen die Berechtigung gewähren,

Aktionen mit den Ressourcen durchzuführen, die sie benötigen. Der Administrator muss diese Richtlinien anschließend den Benutzern anfügen, die sie benötigen.

Informationen zum Erstellen einer IAM identitätsbasierten Richtlinie mithilfe dieser Beispieldokumente zu JSON-Richtlinien finden Sie unter IAM Richtlinien erstellen im IAM Benutzerhandbuch.

Einzelheiten zu Aktionen und Ressourcentypen, die von Verified Permissions definiert wurden, einschließlich des Formats ARNs für die einzelnen Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Verified Permissions in der Service Authorization Reference.

Themen

- Bewährte Methoden für Richtlinien
- Verwenden der Konsole „Verified Permissions“
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand Ressourcen mit verifizierten Berechtigungen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie AWS im IAM Benutzerhandbuch unter AWS Verwaltete Richtlinien oder Verwaltete Richtlinien für Jobfunktionen.
- Berechtigungen mit den geringsten Rechten anwenden — Wenn Sie Berechtigungen mit IAM Richtlinien festlegen, gewähren Sie nur die Berechtigungen, die für die Ausführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung IAM zum Anwenden von Berechtigungen finden Sie IAM im Benutzerhandbuch unter Richtlinien und Berechtigungen.IAM

- Verwenden Sie Bedingungen in IAM Richtlinien, um den Zugriff weiter einzuschränken — Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen einzuschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese im Rahmen einer bestimmten Aktion verwendet werden AWS-Service, wie AWS CloudFormation z. Weitere Informationen finden Sie unter IAM -JSON-Richtlinienelemente: Bedingung im IAM - Benutzerhandbuch.
- Verwenden Sie IAM Access Analyzer, um Ihre IAM Richtlinien zu validieren, um sichere und funktionale Berechtigungen zu gewährleisten. IAM Access Analyzer validiert neue und bestehende Richtlinien, sodass die Richtlinien der IAM Richtliniensprache (JSON) und den IAM-Best Practices entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie im Benutzerhandbuch unter Überprüfen von Richtlinien mit IAM Access Analyzer.IAM
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden finden Sie unter Bewährte Sicherheitsmethoden IAM im IAM Benutzerhandbuch. IAM

Verwenden der Konsole „Verified Permissions“

Um auf die Amazon Verified Permissions-Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, die Ressourcen mit verifizierten Berechtigungen in Ihrem aufzulisten und Details zu diesen Ressourcen anzuzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen die Konsole für verifizierte Berechtigungen weiterhin verwenden können, fügen Sie den Entitäten auch die verifizierten Berechtigungen *ConsoleAccess* oder die *ReadOnly* AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie im Benutzerhandbuch unter Hinzufügen von Berechtigungen für einen IAM Benutzer.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

AWS verwaltete Richtlinien für Amazon Verified Permissions

Um Benutzern, Gruppen und Rollen Berechtigungen hinzuzufügen, ist es einfacher, AWS verwaltete Richtlinien zu verwenden, als Richtlinien selbst zu schreiben. Es erfordert Zeit und Fachwissen, um vom IAM Kunden verwaltete Richtlinien zu erstellen, die Ihrem Team nur die Berechtigungen gewähren, die es benötigt. Um schnell loszulegen, können Sie unsere AWS verwalteten Richtlinien verwenden. Diese Richtlinien decken allgemeine Anwendungsfälle ab und sind in Ihrem AWS-Konto verfügbar. Weitere Informationen zu AWS verwalteten Richtlinien finden Sie im IAM Benutzerhandbuch unter AWS Verwaltete Richtlinien.

AWS Dienste verwalten und aktualisieren AWS verwaltete Richtlinien. Sie können die Berechtigungen in AWS verwalteten Richtlinien nicht ändern. Services fügen einer von AWS verwalteten Richtlinien gelegentlich zusätzliche Berechtigungen hinzu, um neue Features zu unterstützen. Diese Art von Update betrifft alle Identitäten (Benutzer, Gruppen und Rollen), an welche die Richtlinie angehängt ist. Services aktualisieren eine von AWS verwaltete Richtlinie am ehesten, ein neues Feature gestartet wird oder neue Vorgänge verfügbar werden. Dienste entfernen keine Berechtigungen aus einer AWS verwalteten Richtlinie, sodass durch Richtlinienaktualisierungen Ihre bestehenden Berechtigungen nicht beeinträchtigt werden.

AWS Unterstützt außerdem verwaltete Richtlinien für Jobfunktionen, die sich über mehrere Dienste erstrecken. Die ReadOnlyAccess AWS verwaltete Richtlinie bietet beispielsweise schreibgeschützten Zugriff auf alle AWS Dienste und Ressourcen. Wenn ein Dienst eine neue Funktion startet, werden nur Leseberechtigungen für neue Operationen und Ressourcen AWS hinzugefügt. Eine Liste und eine Beschreibung der Richtlinien für Jobfunktionen finden Sie im IAM Benutzerhandbuch unter AWS Verwaltete Richtlinien für Jobfunktionen.

AWS verwaltete Richtlinie: AmazonVerifiedPermissionsFullAccess

Die `AmazonVerifiedPermissionsFullAccess` verwaltete Richtlinie gewährt vollen Zugriff auf verifizierte Berechtigungen. Um mit Amazon Cognito-basierten Identitätsquellen zu arbeiten, müssen Sie eine separate Richtlinie anhängen, z. B. die Richtlinie. AmazonCognitoReadOnly

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:CreatePolicyStore",
        "verifiedpermissions:ListPolicyStores"
      ],
      "Resource": "*"
    },
    {
      "Sid": "PolicyStoreLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:*"
      ],
      "Resource": [
        "arn:aws:verifiedpermissions::*:policy-store/*"
      ]
    }
  ]
}
```

AWS verwaltete Richtlinie: AmazonVerifiedPermissionsReadOnlyAccess

Die `AmazonVerifiedPermissionsReadOnlyAccess` verwaltete Richtlinie gewährt nur Lesezugriff auf verifizierte Berechtigungen.

Diese Richtlinie gewährt Zugriff auf alle Lesevorgänge von Amazon Verified Permissions, einschließlich der Autorisierungsabfrage APIs `IsAuthorized` und`IsAuthorizedWithToken`.

> **Note**
>
> Der Zugriff auf `BatchIsAuthorized` und `BatchIsAuthorizedWithToken` wird automatisch gewährt, wenn der Zugriff auf bzw. gewährt wird. `IsAuthorized` `IsAuthorizedWithToken`

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:ListPolicyStores"
      ],
      "Resource": "*"
    },
    {
      "Sid": "PolicyStoreLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:GetIdentitySource",
        "verifiedpermissions:GetPolicy",
        "verifiedpermissions:GetPolicyStore",
        "verifiedpermissions:GetPolicyTemplate",
        "verifiedpermissions:GetSchema",
        "verifiedpermissions:IsAuthorized",
        "verifiedpermissions:IsAuthorizedWithToken",
        "verifiedpermissions:ListIdentitySources",
        "verifiedpermissions:ListPolicies",
        "verifiedpermissions:ListPolicyTemplates"
      ],
      "Resource": [
        "arn:aws:verifiedpermissions::*:policy-store/*"
      ]
    }
  ]
}
```

Verifizierte Berechtigungen und Aktualisierungen AWS verwalteter Richtlinien

Hier finden Sie Informationen zu Aktualisierungen der AWS verwalteten Richtlinien für verifizierte Berechtigungen, seit dieser Dienst begonnen hat, diese Änderungen nachzuverfolgen. Wenn Sie automatische Benachrichtigungen über Änderungen an dieser Seite erhalten möchten, abonnieren Sie den RSS-Feed auf der Seite mit dem Verlauf des Dokuments mit verifizierten Berechtigungen.

Änderung	Beschreibung	Datum
AmazonVerifiedPermissionsFullAccess – Neue Richtlinie	Verified Permissions hat eine neue Richtlinie hinzugefügt, die vollen Zugriff auf verifizierte Berechtigungen ermöglicht.	11. Oktober 2024
AmazonVerifiedPermissionsReadOnlyAccess – Neue Richtlinie	Verified Permissions hat eine neue Richtlinie hinzugefügt, die den Zugriff auf alle Lesevorgänge von Amazon Verified Permissions ermöglicht, einschließlich der Autorisierungsabfrage APIs `IsAuthorized` und`IsAuthorizedWithToken` .	11. Oktober 2024
Verified Permissions hat begonnen, Änderungen nachzuverfolgen	Verified Permissions hat damit begonnen, Änderungen an den AWS verwalteten Richtlinien nachzuverfolgen.	11. Oktober 2024

Fehlerbehebung bei Identität und Zugriff auf Amazon Verified Permissions

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit verifizierten Berechtigungen und auftreten können IAM.

Themen

- Ich bin nicht berechtigt, eine Aktion unter Verifizierte Berechtigungen auszuführen
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Ressourcen mit verifizierten Berechtigungen ermöglichen

Ich bin nicht berechtigt, eine Aktion unter Verifizierte Berechtigungen auszuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `verifiedpermissions:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 verifiedpermissions:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `verifiedpermissions:`*`GetWidget`*-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion auszuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Verified Permissions übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen Verified Permissions `marymajor` versucht, über die Konsole eine Aktion auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Ressourcen mit verifizierten Berechtigungen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob Verified Permissions diese Funktionen unterstützt, finden Sie unter. So funktioniert Amazon Verified Permissions mit IAM
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im Benutzerhandbuch unter Bereitstellen von Zugriff für einen IAM-Benutzer in einem anderen AWS-Konto , den IAM Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie Zugriff über einen Identitätsverbund gewähren, finden Sie im Benutzerhandbuch unter Zugriff für extern authentifizierte Benutzer (Identitätsverbund).IAM
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontenübergreifenden Zugriff finden Sie IAM im Benutzerhandbuch unter Kontoübergreifender Ressourcenzugriff.IAM

Konformitätsprüfung für von Amazon Verified Permissions

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- Compliance und Governance im Bereich Sicherheit – In diesen Anleitungen für die Lösungsimplementierung werden Überlegungen zur Architektur behandelt. Außerdem werden Schritte für die Bereitstellung von Sicherheits- und Compliance-Features beschrieben.
- Referenz für berechtigte HIPAA-Services – Listet berechtigte HIPAA-Services auf. Nicht alle AWS-Services sind HIPAA-fähig.
- 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. Die Liste der unterstützten Services und Kontrollen finden Sie in der Security-Hub-Steuerelementreferenz.
- 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 bei von Amazon verifizierten Berechtigungen

Die AWS globale Infrastruktur basiert auf AWS-Regionen Availability Zones. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Wenn Sie einen Richtlinienspeicher für verifizierte Berechtigungen erstellen, wird er für eine einzelne Person erstellt und automatisch in den Rechenzentren repliziert AWS-Region, die die Availability Zones dieser Region bilden. Derzeit unterstützt Verified Permissions keine regionsübergreifende Replikation.

Weitere Informationen zu Availability Zones AWS-Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Überwachung von API-Aufrufen von Amazon Verified Permissions

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon Verified Permissions und Ihren anderen AWS Lösungen. AWS bietet die folgenden Tools, um verifizierte Berechtigungen zu überwachen, zu melden, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen zu ergreifen:

- AWS CloudTrailerfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS Kontos getätigt wurden, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können feststellen, welche Benutzer und Konten angerufen wurden AWS, von welcher Quell-IP-Adresse aus die Anrufe getätigt wurden und wann die Aufrufe erfolgten. Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.

Weitere Informationen zur Überwachung von Verified Permissions mit CloudTrail finden Sie unterProtokollieren von API-Aufrufen von Amazon Verified Permissions mit AWS CloudTrail.

Protokollieren von API-Aufrufen von Amazon Verified Permissions mit AWS CloudTrail

Amazon Verified Permissions ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS Dienstes in Verified Permissions bereitstellt. CloudTrail erfasst alle API-Aufrufe für verifizierte Berechtigungen als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Verified Permissions-Konsole und Code-Aufrufe der API-Operationen für verifizierte Berechtigungen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für verifizierte Berechtigungen. Wenn Sie keinen Trail konfigurieren, können Sie in der CloudTrail Konsole im Event-Verlauf trotzdem die neuesten Management-Aktionsereignisse einsehen, aber keine Ereignisse für API-Aufrufe wie`isAuthorized`. Anhand der von CloudTrail gesammelten Informationen können Sie die Anfrage an Verified Permissions, 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 verifizierten Berechtigungen in CloudTrail

CloudTrail ist auf Ihrem aktiviert AWS-Konto , wenn Sie das Konto erstellen. Wenn unter Verifizierte Berechtigungen eine Aktivität auftritt, wird diese Aktivität zusammen mit anderen CloudTrail AWS Dienstereignissen im Ereignisverlauf in einem Ereignis aufgezeichnet. Sie können in Ihrem AWS-Konto die neusten Ereignisse anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit dem CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem Konto AWS-Konto, einschließlich der Ereignisse für verifizierte Berechtigungen, 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 anlegen, gilt dieser für alle AWS-Regionen-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 Aktionen mit verifizierten Berechtigungen werden vom Amazon Verified Permissions API Reference Guide protokolliert CloudTrail und sind im Amazon Verified Permissions API Reference Guide dokumentiert. Beispielsweise generieren Aufrufe der `ListPolicyStores` Aktionen`CreateIdentitySource`,`DeletePolicy`, und Einträge in den CloudTrail Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit 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 CloudTrail -Element userIdentity.

Datenereignisse wie IsAuthorizedund IsAuthorizedWithTokenwerden standardmäßig nicht protokolliert, wenn Sie einen Trail- oder Event-Datenspeicher erstellen. Um CloudTrail Datenereignisse aufzuzeichnen, müssen Sie explizit die unterstützten Ressourcen oder Ressourcentypen hinzufügen, für die Sie Aktivitäten erfassen möchten. Weitere Informationen finden Sie unter Datenereignisse im Benutzerhandbuch für AWS CloudTrail .

Grundlegendes zu Einträgen in der Protokolldatei „Verifizierte

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.

Bei API-Aufrufen zur Autorisierung sind die Antwortelemente, wie z. B. die Entscheidung, `additionalEventData` nicht unter enthaltenresponseElements.

Themen

- IsAuthorized
- BatchIsAuthorized
- CreatePolicyStore
- ListPolicyStores
- DeletePolicyStore
- PutSchema
- GetSchema
- CreatePolicyTemplate
- DeletePolicyTemplate
- CreatePolicy
- GetPolicy
- CreateIdentitySource
- GetIdentitySource
- ListIdentitySources

- DeleteIdentitySource

> **Note**
>
> Aus Datenschutzgründen wurden einige Felder aus den Beispielen geschwärzt.

IsAuthorized


```
{
    "eventVersion": "1.08",
    "userIdentity": {
  "type": "AssumedRole",
  "principalId": "EXAMPLE_PRINCIPAL_ID",
  "arn": "arn:aws:iam::123456789012:role/ExampleRole",
  "accountId": "123456789012",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
    },
    "eventTime": "2023-11-20T22:55:03Z",
    "eventSource": "verifiedpermissions.amazonaws.com",
    "eventName": "IsAuthorized",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "aws-cli/2.11.18 Python/3.11.3 Linux/5.4.241-160.348.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/verifiedpermissions.is-authorized",
    "requestParameters": {
        "principal": {
            "entityType": "PhotoFlash::User",
            "entityId": "alice"
        },
        "action": {
            "actionType": "PhotoFlash::Action",
            "actionId": "ViewPhoto"
        },
        "resource": {
            "entityType": "PhotoFlash::Photo",
            "entityId": "VacationPhoto94.jpg"
        },
        "policyStoreId": "PSEXAMPLEabcdefg111111"
    },
    "responseElements": null,
    "additionalEventData": {
```

```
        "decision": "ALLOW"
    },
    "requestID": "346c4b6a-d12f-46b6-bc06-6c857bd3b28e",
    "eventID": "8a4fed32-9605-45dd-a09a-5ebbf0715bbc",
    "readOnly": true,
    "resources": [
        {
    "accountId": "123456789012",
    "type": "AWS::VerifiedPermissions::PolicyStore",
    "ARN": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data"
}
```


BatchIsAuthorized


```
{
    "eventVersion": "1.08",
    "userIdentity": {
  "type": "AssumedRole",
  "principalId": "EXAMPLE_PRINCIPAL_ID",
  "arn": "arn:aws:iam::123456789012:role/ExampleRole",
  "accountId": "123456789012",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
    },
    "eventTime": "2023-11-20T23:02:33Z",
    "eventSource": "verifiedpermissions.amazonaws.com",
    "eventName": "BatchIsAuthorized",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "aws-cli/2.11.18 Python/3.11.3 Linux/5.4.241-160.348.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/verifiedpermissions.is-authorized",
    "requestParameters": {
        "requests": [
            {
                "principal": {
                    "entityType": "PhotoFlash::User",
                    "entityId": "alice"
```

```
                },
                "action": {
                    "actionType": "PhotoFlash::Action",
                    "actionId": "ViewPhoto"
                },
                "resource": {
                    "entityType": "PhotoFlash::Photo",
                    "entityId": "VacationPhoto94.jpg"
                }
            },
            {
                "principal": {
                    "entityType": "PhotoFlash::User",
                    "entityId": "annalisa"
                },
                "action": {
                    "actionType": "PhotoFlash::Action",
                    "actionId": "DeletePhoto"
                },
                "resource": {
                    "entityType": "PhotoFlash::Photo",
                    "entityId": "VacationPhoto94.jpg"
                }
            }
        ],
        "policyStoreId": "PSEXAMPLEabcdefg111111"
    },
    "responseElements": null,
    "additionalEventData": {
        "results": [
            {
                "request": {
                    "principal": {
                        "entityType": "PhotoFlash::User",
                        "entityId": "alice"
                    },
                    "action": {
                        "actionType": "PhotoFlash::Action",
                        "actionId": "ViewPhoto"
                    },
                    "resource": {
                        "entityType": "PhotoFlash::Photo",
                        "entityId": "VacationPhoto94.jpg"
                    }
```

```
                },
                "decision": "ALLOW"
            },
            {
                "request": {
                    "principal": {
                        "entityType": "PhotoFlash::User",
                        "entityId": "annalisa"
                    },
                    "action": {
                        "actionType": "PhotoFlash::Action",
                        "actionId": "DeletePhoto"
                    },
                    "resource": {
                        "entityType": "PhotoFlash::Photo",
                        "entityId": "VacationPhoto94.jpg"
                    }
                },
                "decision": "DENY"
            }
        ]
    },
    "requestID": "a8a5caf3-78bd-4139-924c-7101a8339c3b",
    "eventID": "7d81232f-f3d1-4102-b9c9-15157c70487b",
    "readOnly": true,
    "resources": [
        {
    "accountId": "123456789012",
    "type": "AWS::VerifiedPermissions::PolicyStore",
    "ARN": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data"
}
```

CreatePolicyStore

```
{
  "eventVersion": "1.08",
```

```
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:33Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreatePolicyStore",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "clientToken": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "validationSettings": {
      "mode": "OFF"
    }
  },
  "responseElements": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
    "createdDate": "2023-05-22T07:43:33.962794Z",
    "lastUpdatedDate": "2023-05-22T07:43:33.962794Z"
  },
  "requestID": "1dd9360e-e2dc-4554-ab65-b46d2cf45c29",
  "eventID": "b6edaeee-3584-4b4e-a48e-311de46d7532",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```


ListPolicyStores


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
```

```
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:33Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "ListPolicyStores",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "maxResults": 10
  },
  "responseElements": null,
  "requestID": "5ef238db-9f87-4f37-ab7b-6cf0ba5df891",
  "eventID": "b0430fb0-12c3-4cca-8d05-84c37f99c51f",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```


DeletePolicyStore


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:32Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "DeletePolicyStore",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
```

```
  "requestID": "1368e8f9-130d-45a5-b96d-99097ca3077f",
  "eventID": "ac482022-b2f6-4069-879a-dd509123d8d7",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

PutSchema

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-16T12:58:57Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "PutSchema",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": {
    "lastUpdatedDate": "2023-05-16T12:58:57.513442Z",
    "namespaces": "[some_namespace]",
    "createdDate": "2023-05-16T12:58:57.513442Z",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
  },
```

```
  "requestID": "631fbfa1-a959-4988-b9f8-f1a43ff5df0d",
  "eventID": "7cd0c677-733f-4602-bc03-248bae581fe5",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "ARN": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

GetSchema

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::222222222222:role/ExampleRole",
    "accountId": "222222222222",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-25T01:12:07Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "GetSchema",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "a1f4d4cd-6156-480a-a9b8-e85a71dcc7c2",
  "eventID": "0b3b8e3d-155c-46f3-a303-7e9e8b5f606b",
  "readOnly": true,
  "resources": [
    {
```

```
      "accountId": "222222222222",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "ARN": "arn:aws:verifiedpermissions::222222222222:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "222222222222",
  "eventCategory": "Management"
}
```

CreatePolicyTemplate

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-16T13:00:24Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreatePolicyTemplate",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": {
    "lastUpdatedDate": "2023-05-16T13:00:23.444404Z",
    "createdDate": "2023-05-16T13:00:23.444404Z",
    "policyTemplateId": "PTEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
  },
  "requestID": "73953bda-af5e-4854-afe2-7660b492a6d0",
  "eventID": "7425de77-ed84-4f91-a4b9-b669181cc57b",
  "readOnly": false,
  "resources": [
    {
```

```
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

DeletePolicyTemplate

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::222222222222:role/ExampleRole",
    "accountId": "222222222222",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-25T01:11:48Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "DeletePolicyTemplate",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "5ff0f22e-6bbd-4b85-a400-4fb74aa05dc6",
  "eventID": "c0e0c689-369e-4e95-a9cd-8de113d47ffa",
  "readOnly": false,
  "resources": [
    {
      "accountId": "222222222222",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "ARN": "arn:aws:verifiedpermissions::222222222222:policy-store/
PSEXAMPLEabcdefg111111"
```

```
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "222222222222",
  "eventCategory": "Management"
}
```

CreatePolicy

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:42:30Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreatePolicy",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "clientToken": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
        "entityType": "PhotoApp::Role",
        "entityId": "PhotoJudge"
    },
    "resource": {
        "entityType": "PhotoApp::Application",
        "entityId": "PhotoApp"
    },
    "lastUpdatedDate": "2023-05-22T07:42:30.70852Z",
    "createdDate": "2023-05-22T07:42:30.70852Z"
```

```
  },
  "requestID": "93ffa151-3841-4960-9af6-30a7f817ef93",
  "eventID": "30ab405f-3dff-43ff-8af9-f513829e8bde",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

GetPolicy

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:29Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "GetPolicy",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyId": "SPEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "23022a9e-2f5c-4dac-b653-59e6987f2fac",
  "eventID": "9b4d5037-bafa-4d57-b197-f46af83fc684",
  "readOnly": true,
```

```
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

CreateIdentitySource

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-19T01:27:44Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreateIdentitySource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "clientToken": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "configuration": {
      "cognitoUserPoolConfiguration": {
        "userPoolArn": "arn:aws:cognito-idp:000011112222:us-east-1:userpool/us-
east-1_aaaaaaaaaa"
      }
    },
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "principalEntityType": "User"
  },
  "responseElements": {
```

```
    "createdDate": "2023-07-14T15:05:01.599534Z",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-07-14T15:05:01.599534Z",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
    },
  "requestID": "afcc1e67-d5a4-4a9b-a74c-cdc2f719391c",
  "eventID": "f13a41dc-4496-4517-aeb8-a389eb379860",
  "readOnly": false,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```

GetIdentitySource

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-24T19:55:31Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "GetIdentitySource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
```

```
  "responseElements": null,
  "requestID": "7a6ecf79-c489-4516-bb57-9ded970279c9",
  "eventID": "fa158e6c-f705-4a15-a731-2cdb4bd9a427",
  "readOnly": true,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```

ListIdentitySources

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-24T20:05:32Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "ListIdentitySources",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "95d2a7bc-7e9a-4efe-918e-97e558aacaf7",
  "eventID": "d3dc53f6-1432-40c8-9d1d-b9eeb75c6193",
  "readOnly": true,
  "resources": [
```

```
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```

DeleteIdentitySource

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-24T19:55:32Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "DeleteIdentitySource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "d554d964-0957-4834-a421-c417bd293086",
  "eventID": "fe4d867c-88ee-4e5d-8d30-2fbc208c9260",
  "readOnly": false,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
```

```
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```

Ressourcen mit Amazon Verified Permissions erstellen mit AWS CloudFormation

Amazon Verified Permissions ist integriert AWS CloudFormation, ein Service, 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. Richtlinienspeicher) und diese Ressourcen für Sie AWS CloudFormation bereitstellt und konfiguriert.

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

> **Important**
>
> Amazon Cognito Identity ist nicht in allen Versionen verfügbar AWS-Regionen wie Amazon Verified Permissions. Wenn Sie eine Fehlermeldung AWS CloudFormation bezüglich Amazon Cognito Identity erhalten, z. B. empfehlen wir Ihnen`Unrecognized resource types: AWS::Cognito::UserPool, AWS::Cognito::UserPoolClient`, den Amazon Cognito-Benutzerpool und den Client in der geografisch nächstgelegenen Region zu erstellen, AWS-Region an der Amazon Cognito Identity verfügbar ist. Verwenden Sie diesen neu erstellten Benutzerpool, wenn Sie die Identitätsquelle Verified Permissions erstellen.

Verifizierte Berechtigungen und AWS CloudFormation Vorlagen

Um Ressourcen für Verified Permissions und verwandte Dienste bereitzustellen und zu konfigurieren, müssen Sie sich mit AWS CloudFormation Vorlagen auskennen. 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.

Verified Permissions unterstützt das Erstellen von Identitätsquellen, Richtlinien, Richtlinienspeichern und Richtlinienvorlagen in AWS CloudFormation. Weitere Informationen, einschließlich Beispielen

für JSON- und YAML-Vorlagen für Ressourcen mit verifizierten Berechtigungen, finden Sie in der Referenz zum Ressourcentyp Amazon Verified Permissions im AWS CloudFormation Benutzerhandbuch.

AWS CDK-Konstrukte

Das AWS Cloud Development Kit (AWS CDK) ist ein Open-Source-Framework für die Softwareentwicklung, mit dem Cloud-Infrastruktur im Code definiert und bereitgestellt werden kann. AWS CloudFormation Konstrukte oder wiederverwendbare Cloud-Komponenten können zum Erstellen von Vorlagen verwendet werden. AWS CloudFormation Diese Vorlagen können dann zur Bereitstellung Ihrer Cloud-Infrastruktur verwendet werden.

Weitere Informationen und das Herunterladen von AWS CDK finden Sie unter AWS Cloud Development Kit.

Im Folgenden finden Sie Links zur Dokumentation für AWS CDK Ressourcen mit verifizierten Berechtigungen, z. B. Konstrukte.

- Von Amazon verifizierte Berechtigungen L2 CDK Construct

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

Greifen Sie mit Amazon Verified Permissions zu AWS PrivateLink

Sie können AWS PrivateLink damit eine private Verbindung zwischen Ihrer VPC und Amazon Verified Permissions herstellen. Sie können auf verifizierte Berechtigungen zugreifen, als ob sie sich in Ihrer VPC befinden würden, ohne ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder AWS Direct Connect eine Verbindung verwenden zu müssen. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um auf verifizierte Berechtigungen zuzugreifen.

Sie stellen diese private Verbindung her, indem Sie einen Schnittstellen-Endpunkt erstellen, der von AWS PrivateLink unterstützt wird. Wir erstellen eine Endpunkt-Netzwerkschnittstelle in jedem Subnetz, das Sie für den Schnittstellen-Endpunkt aktivieren. Dabei handelt es sich um vom Antragsteller verwaltete Netzwerkschnittstellen, die als Einstiegspunkt für Datenverkehr dienen, der für verifizierte Berechtigungen bestimmt ist.

Weitere Informationen finden Sie unter Zugriff auf AWS-Services über AWS PrivateLink im AWS PrivateLink -Leitfaden.

Überlegungen zu verifizierten Berechtigungen

Bevor Sie einen Schnittstellenendpunkt für verifizierte Berechtigungen einrichten, lesen Sie die Überlegungen im AWS PrivateLink Handbuch.

Verified Permissions unterstützt Aufrufe aller API-Aktionen über den Schnittstellenendpunkt.

VPC-Endpunktrichtlinien werden für verifizierte Berechtigungen nicht unterstützt. Standardmäßig ist der vollständige Zugriff auf verifizierte Berechtigungen über den Schnittstellenendpunkt zulässig. Alternativ können Sie den Netzwerkschnittstellen des Endpunkts eine Sicherheitsgruppe zuordnen, um den Datenverkehr zu verifizierten Berechtigungen über den Schnittstellenendpunkt zu kontrollieren.

Erstellen Sie einen Schnittstellenendpunkt für verifizierte Berechtigungen

Sie können einen Schnittstellenendpunkt für Verified Permissions entweder mit der Amazon VPC-Konsole oder mit AWS Command Line Interface (AWS CLI) erstellen. Weitere Informationen finden Sie unter Erstellen eines Schnittstellenendpunkts im AWS PrivateLink -Leitfaden.

Erstellen Sie einen Schnittstellenendpunkt für Verified Permissions mit dem folgenden Servicenamen:

```
com.amazonaws.region.verifiedpermissions
```

Wenn Sie privates DNS für den Schnittstellenendpunkt aktivieren, können Sie API-Anfragen an Verified Permissions stellen, indem Sie den standardmäßigen regionalen DNS-Namen verwenden. Beispiel, `verifiedpermissions.us-east-1.amazonaws.com`.

Erstellen einer Endpunktrichtlinie für Ihren Schnittstellen-Endpunkt

Eine Endpunktrichtlinie ist eine IAM-Ressource, die Sie an einen Schnittstellen-Endpunkt anfügen können. Die standardmäßige Endpunktrichtlinie ermöglicht den vollen Zugriff auf verifizierte Berechtigungen über den Schnittstellenendpunkt. Um den Zugriff auf verifizierte Berechtigungen von Ihrer VPC aus zu kontrollieren, fügen Sie dem Schnittstellenendpunkt eine benutzerdefinierte Endpunktrichtlinie hinzu.

Eine Endpunktrichtlinie gibt die folgenden Informationen an:

- Die Prinzipale, die Aktionen ausführen können (AWS-Konten, IAM-Benutzer und IAM-Rollen).
- Aktionen, die ausgeführt werden können
- Die Ressourcen, auf denen die Aktionen ausgeführt werden können.

Weitere Informationen finden Sie unter Steuern des Zugriffs auf Services mit Endpunktrichtlinien im AWS PrivateLink -Leitfaden.

Beispiel: VPC-Endpunktrichtlinie für Aktionen mit verifizierten Berechtigungen

Im Folgenden finden Sie ein Beispiel für eine benutzerdefinierte Endpunktrichtlinie. Wenn Sie diese Richtlinie an Ihren Schnittstellenendpunkt anhängen, gewährt sie allen Prinzipalen auf allen Ressourcen Zugriff auf die aufgelisteten Aktionen mit verifizierten Berechtigungen.

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "verifiedpermissions:IsAuthorized",
            "verifiedpermissions:IsAuthorizedWithToken",
```

```
            "verifiedpermissions:GetPolicy"
         ],
         "Resource":"*"
      }
   ]
}
```

Kontingente für von Amazon verifizierte Berechtigungen

Ihr AWS-Konto hat Standardkontingente, die früher als Grenzwerte bezeichnet wurden, für jeden AWS Dienst. Wenn nicht anders angegeben, gilt jedes Kontingent spezifisch für eine Region. Sie können Erhöhungen für einige Kontingente beantragen und andere Kontingente können nicht erhöht werden.

Um die Kontingente für verifizierte Berechtigungen anzuzeigen, öffnen Sie die Konsole Service Quotas. Wählen Sie im Navigationsbereich AWS Dienste und dann Verifizierte Berechtigungen aus.

Informationen zur Erhöhung eines Kontingents finden Sie unter Anfordern einer Kontingenterhöhung im Benutzerhandbuch zu Service Quotas. Wenn das Kontingent unter Service Quotas noch nicht in verfügbar ist, verwenden Sie das Formular zur Erhöhung des Service-Limits.

Für Ihr AWS-Konto gelten die folgenden Kontingente in Bezug auf verifizierte Berechtigungen.

Themen

- Kontingente für Ressourcen
- Kontingente für Hierarchien
- Kontingente für Operationen pro Sekunde

Kontingente für Ressourcen

Name	Standard	Anpas	Beschreibung
Policy-Shops pro Region und Konto	Jede unterstützte Region: 30.000	Ja	Die maximale Anzahl von Policy-Stores.
Richtlinienvorlagen pro Richtlinienspeicher	Jede unterstützte Region: 40	Ja	Die maximale Anzahl von Richtlinienvorlagen in einem Richtlinienspeicher.
Identitätsquellen pro Richtlinienspeicher	1	Nein	Die maximale Anzahl von Identitätsquellen, die Sie für einen Richtlini

Name	Standard	Anpas	Beschreibung
			enspeicher definieren können.
Größe der Autorisierungsanforderung[1]	1 MB	Nein	Die maximale Größe einer Autorisierungsanfr age.
Größe der Richtlinie	10,000 Bytes	Nein	Die maximale Größe einer einzelnen Police.
Größe des Schemas	200.000 Byte	Nein	Die maximale Größe des Schemas eines Richtlini enspeichers.
Richtliniengröße pro Ressource	200.000 Byte[2]	Ja	Die maximale Größe aller Richtlinien, die auf eine bestimmte Ressource verweisen.

[1] Das Kontingent für eine Autorisierungsanfrage ist für IsAuthorizedsowohl als auch dasselbe IsAuthorizedWithToken.

[2] Das Standardlimit für die Gesamtgröße aller Richtlinien, die für eine einzelne Ressource gelten, beträgt 200.000 Byte. In ähnlicher Weise ist die Gesamtgröße aller Richtlinien, bei denen der Geltungsbereich die Ressource undefiniert lässt und somit für alle Ressourcen gilt, standardmäßig auf 200.000 Byte begrenzt. Beachten Sie, dass bei Richtlinien, die mit Vorlagen verknüpft sind, die Größe der Richtlinienvorlage nur einmal gezählt wird, zuzüglich der Größe jedes Parametersatzes, der zur Instanziierung jeder mit der Vorlage verknüpften Richtlinie verwendet wird. Dieses Limit kann erhöht werden, sofern Ihr Richtlinienentwurf bestimmte Einschränkungen erfüllt. Wenn Sie diese Option prüfen möchten, wenden Sie sich an Support.

Beispiel für die Größe einer Richtlinie, die mit einer Vorlage verknüpft ist

Sie können ermitteln, wie vorlagenverknüpfte Richtlinien zur Richtliniengröße pro Ressourcenkontingent beitragen, indem Sie die Summe der Länge von Prinzipal und Ressource heranziehen. Wenn der Hauptbenutzer oder die Ressource nicht angegeben ist, ist die Länge dieses

Teils 0. Wenn eine Ressource nicht angegeben ist, wird ihre Größe auf das `"unspecified"` Ressourcenkontingent angerechnet. Die Größe des Vorlagentexts selbst hat keinen Einfluss auf die Größe der Richtlinie.

Schauen wir uns die folgende Vorlage an:

```
@id("template1")
permit (
  principal in ?principal,
  action in [Action::"view", Action::"comment"],
  resource in ?resource
)
unless {
  resource.tag =="private"
};
```

Lassen Sie uns anhand dieser Vorlage die folgenden Richtlinien erstellen:

```
TemplateLinkedPolicy {
  policyId: "policy1",
  templateId: "template1",
  principal: User::"alice",
  resource: Photo::"car.jpg"
}

TemplateLinkedPolicy {
  policyId: "policy2",
  templateId: "template1",
  principal: User::"bob",
  resource: Photo::"boat.jpg"
}

TemplateLinkedPolicy {
  policyId: "policy3",
  templateId: "template1",
  principal: User::"jane",
  resource: Photo::"car.jpg"

TemplateLinkedPolicy {
  policyId: "policy4",
  templateId: "template1",
  principal: User::"jane",
  resource
```

```
}
```

Lassen Sie uns nun die Größe dieser Richtlinien berechnen, indem wir `resource` für jede einzelne die Zeichen im `principal` und zählen. Jedes Zeichen zählt als 1 Byte.

Die Größe von `policy1` wäre die Länge des Prinzipals `User::"alice"` (13) plus die Länge der Ressource `Photo::"car.jpg"` (16). Wenn wir sie zusammenzählen, haben wir 13 + 16 = 29 Byte.

Die Größe von `policy2` wäre die Länge des Prinzipals `User::"bob"` (11) plus die Länge der Ressource `Photo::"boat.jpg"` (17). Wenn wir sie zusammenzählen, haben wir 11 + 17 = 28 Byte.

Die Größe von `policy3` wäre die Länge des Prinzipals `User::"jane"` (12) plus die Länge der Ressource `Photo::"car.jpg"` (16). Wenn wir sie zusammenzählen, haben wir 12 + 16 = 28 Byte.

Die Größe von `policy4` wäre die Länge des Prinzipals `User::"jane"` (12) plus die Länge der Ressource (0). Wenn wir sie zusammenzählen, haben wir 12 + 0 = 12 Byte.

Da dies die einzige Richtlinie `policy2` ist, die auf die Ressource verweistPhoto::"boat.jpg", beträgt die Gesamtressourcengröße 28 Byte.

Da `policy1` und `policy3` beide auf die Ressource verweisenPhoto::"car.jpg", beträgt die Gesamtressourcengröße 29 + 28 = 57 Byte.

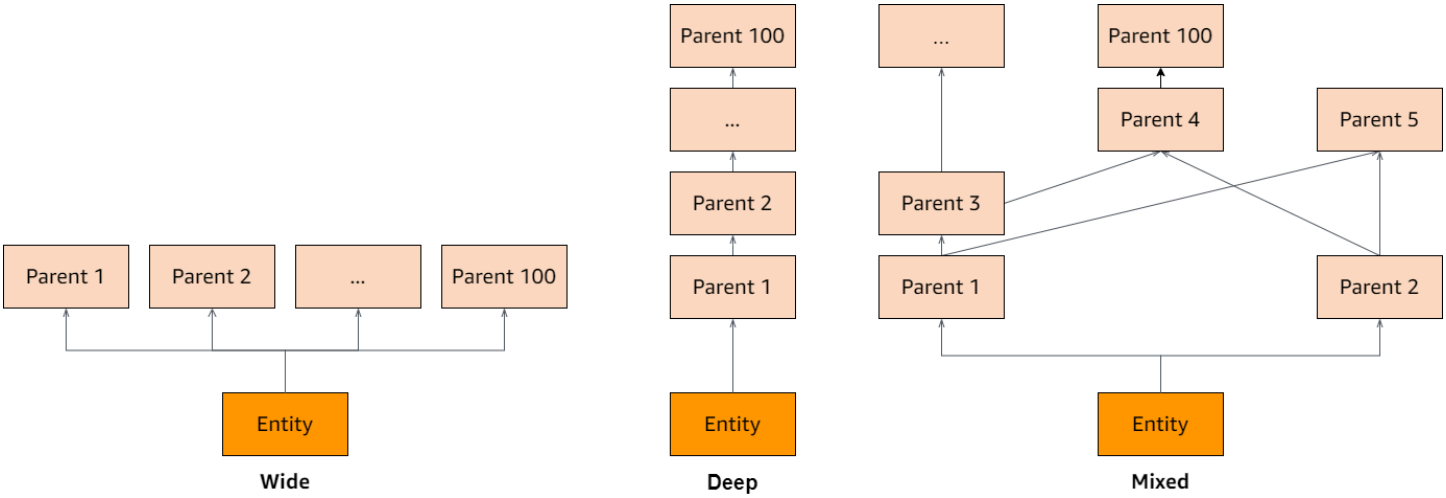
Da dies die einzige Richtlinie `policy4` ist, die auf die `"unspecified"` Ressource verweist, beträgt die Gesamtressourcengröße 12 Byte.

Kontingente für Hierarchien

> **Note**
>
> Die folgenden Kontingente sind aggregiert, d. h. sie werden zusammengezählt. Die maximale Anzahl transitiver Eltern für die Gruppe ist aufgeführt. Wenn die Obergrenze für transitive Eltern pro Schulleiter beispielsweise 100 beträgt, bedeutet das, dass es 100 Eltern von Schulleitern und 0 Eltern sowohl für Aktionen als auch für Ressourcen geben könnte, oder eine beliebige Kombination von Eltern, die zusammen 100 Eltern ergibt.

Name	Standard	Anpas	Beschreibung
Transitive Eltern pro Schulleiter	100	Nein	Die maximale Anzahl transitiver Eltern für jeden Schulleiter.
Transitive Eltern pro Aktion	100	Nein	Die maximale Anzahl transitiver Eltern für jede Aktion.
Transitive Eltern pro Ressource	100	Nein	Die maximale Anzahl transitiver Eltern für jede Ressource.

Das folgende Diagramm zeigt, wie transitive Eltern für eine Entität (Prinzipal, Aktion oder Ressource) definiert werden können.



Kontingente für Operationen pro Sekunde

Verified Permissions drosselt Anfragen an Dienstendpunkte, AWS-Region wenn Anwendungsanfragen das Kontingent für einen API-Vorgang überschreiten. Verified Permissions gibt möglicherweise eine Ausnahme zurück, wenn Sie das Kontingent an Anfragen pro Sekunde überschreiten oder wenn Sie versuchen, gleichzeitig Schreibvorgänge auszuführen. Sie können Ihre aktuellen RPS-Kontingente unter Service Quotas einsehen. Um zu verhindern, dass Anwendungen das Kontingent für einen Vorgang überschreiten, müssen Sie sie für Wiederholungsversuche und

exponentielle Backups optimieren. Weitere Informationen finden Sie unter Wiederholen mit Backoff-Muster und Verwalten und Überwachen der API-Drosselung in Ihren Workloads.

Name	Standard	Anpas	Beschreibung
BatchGetPolicy Anfragen pro Sekunde, Region, pro Konto	Jede unterstützte Region: 10	Yes (Ja)	Die maximale Anzahl von BatchGetPolicy Anfragen pro Sekunde.
BatchIsAuthorized Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 30	Ja	Die maximale Anzahl von BatchIsAuthorized Anfragen pro Sekunde.
BatchIsAuthorizedWithToken Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 30	Yes (Ja)	Die maximale Anzahl von BatchIsAuthorizedWithToken Anfragen pro Sekunde.
CreateIdentitySource Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 1	Ja	Die maximale Anzahl von CreateIdentitySource Anfragen pro Sekunde.
CreatePolicy Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von CreatePolicy Anfragen pro Sekunde.
CreatePolicyStore Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 1	Nein	Die maximale Anzahl von CreatePolicyStore Anfragen pro Sekunde.
CreatePolicyTemplate Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von CreatePolicyTemplate Anfragen pro Sekunde.
DeleteIdentitySource Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 1	Ja	Die maximale Anzahl von DeleteIdentitySource Anfragen pro Sekunde.

Name	Standard	Anpas	Beschreibung
DeletePolicy Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von DeletePolicy Anfragen pro Sekunde.
DeletePolicyStore Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 1	Nein	Die maximale Anzahl von DeletePolicyStore Anfragen pro Sekunde.
DeletePolicyTemplate Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von DeletePolicyTemplate Anfragen pro Sekunde.
GetIdentitySource Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von GetIdentitySource Anfragen pro Sekunde.
GetPolicy Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von GetPolicy Anfragen pro Sekunde.
GetPolicyStore Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von GetPolicyStore Anfragen pro Sekunde.
GetPolicyTemplate Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von GetPolicyTemplate Anfragen pro Sekunde.
GetSchema Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von GetSchema Anfragen pro Sekunde.
IsAuthorized Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 200	Ja	Die maximale Anzahl von IsAuthorized Anfragen pro Sekunde.

Name	Standard	Anpas	Beschreibung
IsAuthorizedWithToken Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 200	Ja	Die maximale Anzahl von IsAuthorizedWithToken Anfragen pro Sekunde.
ListIdentitySources Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von ListIdentitySources Anfragen pro Sekunde.
ListPolicies Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von ListPolicies Anfragen pro Sekunde.
ListPolicyStores Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von ListPolicyStores Anfragen pro Sekunde.
ListPolicyTemplates Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von ListPolicyTemplates Anfragen pro Sekunde.
PutSchema Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von PutSchema Anfragen pro Sekunde.
UpdateIdentitySource Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 1	Ja	Die maximale Anzahl von UpdateIdentitySource Anfragen pro Sekunde.
UpdatePolicy Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von UpdatePolicy Anfragen pro Sekunde.
UpdatePolicyStore Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Nein	Die maximale Anzahl von UpdatePolicyStore Anfragen pro Sekunde.

Name	Standard	Anpas	Beschreibung
UpdatePolicyTemplate Anfragen pro Sekunde, pro Region, pro Konto	Jede unterstützte Region: 10	Ja	Die maximale Anzahl von UpdatePolicyTemplate Anfragen pro Sekunde.

Sprachbegriffe und Konzepte der Amazon Verified Permissions- und Cedar-Richtlinien

Sie sollten die folgenden Konzepte verstehen, um Amazon Verified Permissions zu verwenden.

Konzepte für verifizierte Berechtigungen

- Autorisierungsmodell
- Autorisierungsanfrage
- Antwort auf die Autorisierung
- Überlegte Richtlinien
- Kontextdaten
- Festlegung von Richtlinien
- Daten der Entität
- Berechtigungen, Autorisierung und Prinzipale
- Durchsetzung von Richtlinien
- Richtlinienspeicher
- Zufriedene Richtlinien
- Unterschiede zwischen Amazon Verified Permissions und der Sprache der Cedar-Richtlinie

Sprachkonzepte von Cedar Policy

- Autorisierung
- Entität
- Gruppen und Hierarchien
- Namespaces
- Richtlinie
- Vorlage für eine Richtlinie
- Schema

Autorisierungsmodell

Das Autorisierungsmodell beschreibt den Umfang der von der Anwendung gestellten Autorisierungsanfragen und die Grundlage für die Bewertung dieser Anfragen. Es wird anhand der verschiedenen Ressourcentypen, der mit diesen Ressourcen ergriffenen Maßnahmen und der Art der Hauptpersonen, die diese Aktionen ausführen, definiert. Dabei wird auch der Kontext berücksichtigt, in dem diese Maßnahmen ergriffen werden.

Bei der rollenbasierten Zugriffskontrolle (RBAC) handelt es sich um eine Bewertungsgrundlage, auf der Rollen definiert und mit einer Reihe von Berechtigungen verknüpft werden. Diese Rollen können dann einer oder mehreren Identitäten zugewiesen werden. Die zugewiesene Identität erwirbt die mit der Rolle verknüpften Berechtigungen. Wenn die mit der Rolle verknüpften Berechtigungen geändert werden, wirkt sich die Änderung automatisch auf alle Identitäten aus, denen die Rolle zugewiesen wurde. Cedar kann RBAC-Entscheidungen mithilfe von Hauptgruppen unterstützen.

Bei der attributebasierten Zugriffskontrolle (ABAC) handelt es sich um eine Bewertungsgrundlage, bei der die mit einer Identität verknüpften Berechtigungen anhand der Attribute dieser Identität bestimmt werden. Cedar kann ABAC-Entscheidungen durch die Verwendung von Richtlinienbedingungen unterstützen, die auf die Attribute des Auftraggebers verweisen.

Die Cedar-Richtliniensprache ermöglicht die Kombination von RBAC und ABAC in einer einzigen Richtlinie, indem Berechtigungen für eine Gruppe von Benutzern definiert werden können, für die attributbasierte Bedingungen gelten.

Autorisierungsanfrage

Eine Autorisierungsanfrage ist eine Anforderung verifizierter Berechtigungen durch eine Anwendung, um eine Reihe von Richtlinien auszuwerten, um festzustellen, ob ein Principal in einem bestimmten Kontext eine Aktion an einer Ressource ausführen darf.

Antwort auf die Autorisierung

Die Autorisierungsantwort ist die Antwort auf die Autorisierungsanfrage. Sie enthält eine Entscheidung über das Zulassen oder Verweigern sowie zusätzliche Informationen, z. B. die IDs zu den maßgeblichen Richtlinien.

Überlegte Richtlinien

In Betracht gezogene Richtlinien sind alle Richtlinien, die von Verified Permissions ausgewählt und bei der Bewertung einer Autorisierungsanfrage berücksichtigt werden sollen.

Kontextdaten

Kontextdaten sind Attributwerte, die zusätzliche Informationen zur Auswertung bereitstellen.

Festlegung von Richtlinien

Entscheidende Richtlinien sind die Richtlinien, die die Reaktion auf die Autorisierung bestimmen. Wenn es beispielsweise zwei erfüllte Richtlinien gibt, von denen die eine die Option „Verweigern“ und die andere eine „Zulassen“ ist, dann ist die Ablehnungsrichtlinie die ausschlaggebende Richtlinie. Wenn es mehrere erfüllte Genehmigungsrichtlinien und keine erfüllten Verbotsrichtlinien gibt, dann gibt es mehrere ausschlaggebende Richtlinien. Für den Fall, dass keine Richtlinien übereinstimmen und die Antwort „Verweigert“ lautet, gibt es keine ausschlaggebenden Richtlinien.

Daten der Entität

Entitätsdaten sind Daten über den Prinzipal, die Aktion und die Ressource. Entitätsdaten, die für die Bewertung von Richtlinien relevant sind, umfassen die Gruppenzugehörigkeit bis nach oben in der Entitätshierarchie und die Attributwerte des Prinzipals und der Ressource.

Berechtigungen, Autorisierung und Prinzipale

Verified Permissions verwaltet detaillierte Berechtigungen und Autorisierungen innerhalb von benutzerdefinierten Anwendungen, die Sie erstellen.

Ein Principal ist ein Benutzer einer Anwendung, entweder eines Menschen oder einer Maschine, deren Identität an eine Kennung wie einen Benutzernamen oder eine Computer-ID gebunden ist. Der Authentifizierungsprozess bestimmt, ob der Principal wirklich die Identität ist, für die er sich ausgibt.

Dieser Identität sind eine Reihe von Anwendungsberechtigungen zugeordnet, die festlegen, welche Aktionen dieser Prinzipal in dieser Anwendung ausführen darf. Bei der Autorisierung werden diese Berechtigungen bewertet, um festzustellen, ob ein Hauptbenutzer eine bestimmte Aktion in der Anwendung ausführen darf. Diese Berechtigungen können als Richtlinien ausgedrückt werden.

Durchsetzung von Richtlinien

Die Durchsetzung von Richtlinien ist der Prozess, bei dem die Bewertungsentscheidung innerhalb der Anwendung außerhalb verifizierter Genehmigungen durchgesetzt wird. Wenn bei der Bewertung „Verified Permissions“ der Wert „Verweigert“ zurückgegeben wird, würde die Durchsetzung sicherstellen, dass der Hauptbenutzer am Zugriff auf die Ressource gehindert wird.

Richtlinienspeicher

Ein Richtlinienspeicher ist ein Container für Richtlinien und Vorlagen. Jeder Store enthält ein Schema, das zur Validierung der dem Store hinzugefügten Richtlinien verwendet wird. Standardmäßig hat jede Anwendung ihren eigenen Richtlinienspeicher, aber mehrere Anwendungen können sich einen einzigen Richtlinienspeicher teilen. Wenn eine Anwendung eine Autorisierungsanfrage stellt, identifiziert sie den Richtlinienspeicher, der zur Auswertung dieser Anfrage verwendet wurde. Richtlinienspeicher bieten die Möglichkeit, eine Reihe von Richtlinien zu isolieren. Sie können daher in einer Mehrmandantenanwendung verwendet werden, um die Schemas und Richtlinien für jeden Mandanten zu speichern. Eine einzelne Anwendung kann separate Richtlinienspeicher für jeden Mandanten haben.

Bei der Bewertung einer Autorisierungsanfrage berücksichtigt Verified Permissions nur die Teilmenge der Richtlinien im Richtlinienspeicher, die für die Anfrage relevant sind. Die Relevanz wird anhand des Geltungsbereichs der Richtlinie bestimmt. Der Geltungsbereich gibt den spezifischen Prinzipal und die Ressource an, für die die Richtlinie gilt, sowie die Aktionen, die der Prinzipal mit der Ressource ausführen kann. Die Definition des Geltungsbereichs trägt zur Verbesserung der Leistung bei, da die Anzahl der in Betracht gezogenen Richtlinien eingegrenzt wird.

Zufriedene Richtlinien

Erfüllte Richtlinien sind die Richtlinien, die den Parametern der Autorisierungsanfrage entsprechen.

Unterschiede zwischen Amazon Verified Permissions und der Sprache der Cedar-Richtlinie

Amazon Verified Permissions verwendet die Cedar Policy Language Engine, um seine Autorisierungsaufgaben auszuführen. Es gibt jedoch einige Unterschiede zwischen der systemeigenen Cedar-Implementierung und der Implementierung von Cedar in Verified Permissions. In diesem Thema werden diese Unterschiede beschrieben.

Namespace-Definition

Die Implementierung von Verified Permissions von Cedar unterscheidet sich von der nativen Cedar-Implementierung wie folgt:

- Verified Permissions unterstützt nur einen Namespace in einem Schema, das in einem Richtlinienspeicher definiert ist.
- Mit Verified Permissions können Sie keinen Namespace erstellen, der aus einer leeren Zeichenfolge besteht oder die folgenden Werte enthält: `awsamazon`, oder. `cedar`

Unterstützung von Richtlinienvorlagen

Sowohl Verified Permissions als auch Cedar erlauben Platzhalter im Gültigkeitsbereich nur für `principal` und`resource`. Verified Permissions setzt jedoch auch voraus, dass weder die noch die `principal` Rechte `resource` uneingeschränkt sind.

Die folgende Richtlinie ist in Cedar gültig, wird jedoch von Verified Permissions abgelehnt, da sie nicht eingeschränkt `principal` ist.

```
permit(principal, action == Action::"view", resource == ?resource);
```

Die beiden folgenden Beispiele sind sowohl für Cedar als auch für Verified Permissions gültig, da sowohl für als auch für Verified Permissions `principal` Einschränkungen `resource` gelten.

```
permit(principal == User::"alice", action == Action::"view", resource == ?resource);
```

```
permit(principal == ?principal, action == Action::"a", resource in ?resource);
```

Schema-Unterstützung

Verified Permissions erfordert, dass alle JSON-Schlüsselnamen des Schemas keine leeren Zeichenfolgen sind. Cedar erlaubt in einigen Fällen leere Zeichenfolgen, z. B. für Eigenschaften oder Namespaces.

Definition von Aktionsgruppen

Die Autorisierungsmethoden von Cedar erfordern eine Liste der Entitäten, die bei der Bewertung einer Autorisierungsanfrage anhand der Richtlinien berücksichtigt werden müssen.

Sie können die Aktionen und Aktionsgruppen, die von Ihrer Anwendung verwendet werden, im Schema definieren. Cedar nimmt das Schema jedoch nicht als Teil einer Evaluierungsanfrage auf. Stattdessen verwendet Cedar das Schema nur, um die von Ihnen eingereichten Richtlinien und Richtlinienvorlagen zu validieren. Da Cedar bei Bewertungsanfragen nicht auf das Schema verweist, selbst wenn Sie Aktionsgruppen im Schema definiert haben, müssen Sie auch die Liste aller Aktionsgruppen als Teil der Entitätenliste angeben, die Sie an die Autorisierungs-API-Operationen übergeben müssen.

Verified Permissions erledigt das für Sie. Alle Aktionsgruppen, die Sie in Ihrem Schema definieren, werden automatisch an die Entitätsliste angehängt, an die Sie als Parameter für die `IsAuthorized` `IsAuthorizedWithToken` OR-Operationen übergeben.

Formatierung von Entitäten

Die JSON-Formatierung von Entitäten in Verified Permissions, die den `entityList` Parameter verwenden, unterscheidet sich in folgenden Punkten von Cedar:

- In Verified Permissions müssen bei einem JSON-Objekt alle Schlüssel-Wert-Paare in ein JSON-Objekt mit dem Namen eingeschlossen sein. `Record`
- Eine JSON-Liste in Verified Permissions muss in ein JSON-Schlüssel-Wert-Paar eingeschlossen werden, wobei der Schlüsselname `Set` und der Wert die ursprüngliche JSON-Liste von Cedar sind.
- Für `StringLong`, und `Boolean` Typnamen wird jedes Schlüssel-Wert-Paar aus Cedar in Verified Permissions durch ein JSON-Objekt ersetzt. Der Name des Objekts ist der ursprüngliche Schlüsselname. Innerhalb des JSON-Objekts gibt es ein Schlüssel-Wert-Paar, wobei der Schlüsselname der Typname des Skalarwerts (`StringLong`, oder`Boolean`) und der Wert der Wert aus der Cedar-Entität ist.
- Die Syntaxformatierung von Cedar-Entitäten und Verified Permissions-Entitäten unterscheidet sich in folgenden Punkten:

Cedar-Format	Format für verifizierte Berechtigungen
`uid`	`Identifier`
`type`	`EntityType`
`id`	`EntityId`
`attrs`	`Attributes`

Cedar-Format	Format für verifizierte Berechtigungen
`parents`	`Parents`

Example - Listen

Die folgenden Beispiele zeigen, wie eine Liste von Entitäten in Cedar bzw. Verified Permissions ausgedrückt wird.

Cedar

```
[
  {
    "number": 1
  },
  {
    "sentence": "Here is an example sentence"
  },
  {
    "Question": false
  }
]
```

Verified Permissions

```
{
  "Set": [
    {
      "Record": {
        "number": {
          "Long": 1
        }
      }
    },
    {
      "Record": {
        "sentence": {
          "String": "Here is an example sentence"
        }
      }
    },
```

```
    {
      "Record": {
        "question": {
          "Boolean": false
        }
      }
    }
  ]
}
```

Example - Bewertung der Politik

Die folgenden Beispiele zeigen, wie Entitäten für die Auswertung einer Richtlinie in einer Autorisierungsanfrage in Cedar bzw. Verified Permissions formatiert werden.

Cedar

```
[
    {
        "uid": {
            "type": "PhotoApp::User",
            "id": "alice"
        },
        "attrs": {
            "age": 25,
            "name": "alice",
            "userId": "123456789012"
        },
        "parents": [
            {
                "type": "PhotoApp::UserGroup",
                "id": "alice_friends"
            },
            {
                "type": "PhotoApp::UserGroup",
                "id": "AVTeam"
            }
        ]
    },
    {
        "uid": {
            "type": "PhotoApp::Photo",
```

```
            "id": "vacationPhoto.jpg"
        },
        "attrs": {
            "private": false,
            "account": {
                "__entity": {
                    "type": "PhotoApp::Account",
                    "id": "ahmad"
                }
            }
        },
        "parents": []
    },
    {
        "uid": {
            "type": "PhotoApp::UserGroup",
            "id": "alice_friends"
        },
        "attrs": {},
        "parents": []
    },
    {
        "uid": {
            "type": "PhotoApp::UserGroup",
            "id": "AVTeam"
        },
        "attrs": {},
        "parents": []
    }
]
```

Verified Permissions

```
[
    {
        "Identifier": {
            "EntityType": "PhotoApp::User",
            "EntityId": "alice"
        },
        "Attributes": {
            "age": {
                "Long": 25
            },
```

```
            "name": {
                "String": "alice"
            },
            "userId": {
                "String": "123456789012"
            }
        },
        "Parents": [
            {
                "EntityType": "PhotoApp::UserGroup",
                "EntityId": "alice_friends"
            },
            {
                "EntityType": "PhotoApp::UserGroup",
                "EntityId": "AVTeam"
            }
        ]
    },
    {
        "Identifier": {
            "EntityType": "PhotoApp::Photo",
            "EntityId": "vacationPhoto.jpg"
        },
        "Attributes": {
            "private": {
                "Boolean": false
            },
            "account": {
                "EntityIdentifier": {
                    "EntityType": "PhotoApp::Account",
                    "EntityId": "ahmad"
                }
            }
        },
        "Parents": []
    },
    {
        "Identifier": {
            "EntityType": "PhotoApp::UserGroup",
            "EntityId": "alice_friends"
        },
        "Parents": []
    },
    {
```

```
        "Identifier": {
            "EntityType": "PhotoApp::UserGroup",
            "EntityId": "AVTeam"
        },
        "Parents": []
    }
]
```

Längen- und Größenbeschränkungen

Verified Permissions unterstützt die Speicherung in Form von Richtlinienspeichern für Ihr Schema, Ihre Richtlinien und Richtlinienvorlagen. Aufgrund dieser Speicherung legt Verified Permissions einige Längen- und Größenbeschränkungen fest, die für Cedar nicht relevant sind.

Object	Limit für verifizierte Berechtigungen (in Byte)	Zedernholzlimit
Größe der Police[1]	10.000	Keine
Beschreibung der Online-Richtlinie	150	Gilt nicht für Cedar
Größe der Richtlinienvorlage	10.000	Keine
Größe des Schemas	100 000	Keine
Entitätstyp	200	Keine
Policy ID (Richtlinien-ID)	64	Keine
ID der Richtlinienvorlage	64	Keine
Entity-ID	200	Keine
ID des Richtlinienspeichers	64	Gilt nicht für Cedar

[1] Unter Verifizierte Berechtigungen gibt es eine Obergrenze für Richtlinien pro Richtlinienspeicher, die auf der Gesamtgröße der im Richtlinienspeicher erstellten Richtlinien, Aktionen und Ressourcen basiert. Die Gesamtgröße aller Richtlinien, die sich auf eine einzelne Ressource beziehen, darf

200.000 Byte nicht überschreiten. Bei Richtlinien, die mit Vorlagen verknüpft sind, wird die Größe der Richtlinienvorlage nur einmal gezählt, zuzüglich der Größe jedes Parametersatzes, der zur Instanziierung jeder mit der Vorlage verknüpften Richtlinie verwendet wird.

Häufig gestellte Fragen zum Upgrade von Amazon Verified Permissions auf Cedar v4

Amazon Verified Permissions wird derzeit auf Cedar v4 aktualisiert. Wir arbeiten daran, dies für Sie so reibungslos wie möglich zu gestalten. Im Folgenden FAQs sollten Ihre Fragen beantwortet und Ihnen bei der Vorbereitung geholfen werden.

Themen

- Wie ist der aktuelle Stand des Upgrades?
- Muss ich jetzt etwas tun?
- Wirkt sich das Upgrade der Konsole auf den Autorisierungsdienst aus?
- Was sind die wichtigsten Änderungen in Cedar v3 und Cedar v4?
- Wann wird das Upgrade auf Cedar v4 abgeschlossen sein?

Wie ist der aktuelle Stand des Upgrades?

Als ersten Schritt haben wir die Konsole auf Cedar v4.3 aktualisiert, das Backend läuft jedoch immer noch auf Cedar v2.5.0. Das bedeutet, dass Sie jetzt zwar die Konsole verwenden können, um Richtlinien mit neuen Funktionen wie dem `is` Operator zu erstellen, aber wenn Sie versuchen, sie zu speichern, erhalten Sie immer noch eine Fehlermeldung, bis wir das Upgrade abgeschlossen haben.

Muss ich jetzt etwas tun?

Nein. Sie können mit der Konsole anfangen, Cedar v4 zu erkunden, wenn Sie möchten, müssen aber nichts tun.

Wirkt sich das Upgrade der Konsole auf den Autorisierungsdienst aus?

Nein. Vor dem Upgrade werden wir Tests durchführen, um zu überprüfen, ob Ihr Policy Store mit Cedar v4 ordnungsgemäß funktioniert. Es gibt einige geringfügige grundlegende Änderungen zwischen Version 2.5.0 und Version 4.3, aber es ist sehr unwahrscheinlich, dass Ihr Policy Store davon betroffen ist. Ist dies der Fall, wird Ihr Policy Store nicht aktualisiert und autorisiert weiterhin mit

Cedar v2.5.0. Sollte dies der Fall sein, werden wir uns mit Ihnen in Verbindung setzen, um Ihnen alle Änderungen zu erläutern, die Sie vornehmen müssen, bevor Sie ein Upgrade durchführen können.

Was sind die wichtigsten Änderungen in Cedar v3 und Cedar v4?

Wichtige Änderungen werden im Cedar-Änderungsprotokoll identifiziert, das mit einem gekennzeichnet ist(*).

> **Note**
>
> Wenn Ihr Policy Store von wichtigen Änderungen betroffen ist, wird er nicht aktualisiert, und wir werden mit Ihnen zusammenarbeiten, um den Policy Store zu aktualisieren, damit er aktualisiert werden kann.

Wann wird das Upgrade auf Cedar v4 abgeschlossen sein?

Unser Ziel ist es, dass alle Konten bis zum 31. Dezember 2025 aktualisiert werden.

Dokumentenverlauf für das Amazon Verified Permissions User Guide

In der folgenden Tabelle werden die Dokumentationsversionen für Verified Permissions beschrieben.

Änderung	Beschreibung	Datum
Neue AWS verwaltete Richtlinien	Sie können die `AmazonVerifiedPermissionsFullAccess` und die `AmazonVerifiedPermissionsReadOnlyAccess` IAM verwalteten Richtlinien jetzt mit verifizierten Berechtigungen verwenden.	11. Oktober 2024
OIDC-Identitätsquellen	Sie können jetzt Benutzer von OpenID Connect (OIDC) -Identitätsanbietern autorisieren.	8. Juni 2024
Batch-Autorisierung mit Identitätsquellen-Token	Sie können jetzt Benutzer aus einem Amazon Cognito Cognito-Benutzerpool in einer einzigen `BatchIsAuthorizedWithToken` API-Anfrage autorisieren.	5. April 2024
Einen Richtlinienspeicher mit API Gateway erstellen	Sie können jetzt einen Richtlinienspeicher aus einer vorhandenen API und einem Amazon Cognito Cognito-Benutzerpool erstellen.	1. April 2024
Kontextkonzepte und Beispiel	Es wurden Informationen zum Kontext in Autorisierungsanfr	1. Februar 2024

	agen mit verifizierten Berechtig ungen hinzugefügt.	
Autorisierungskonzepte und Beispiel	Es wurden Informationen zu Autorisierungsanfragen mit verifizierten Berechtigungen hinzugefügt.	1. Februar 2024
AWS CloudFormation Integrati on	Verified Permissions unterstüt zt das Erstellen von Identität squellen, Richtlinien, Richtlinienspeichern und Richtlinienvorlagen in AWS CloudFormation.	30. Juni 2023
Erstversion	Erste Version des Amazon Verified Permissions User Guide	13. Juni 2023

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.